

# Package: gofigR (via r-universe)

March 12, 2025

**Type** Package

**Title** Client for 'GoFigr.io'

**Version** 0.3.0

**Description** Integrates with your 'RMarkdown' documents to automatically publish figures to the <https://GoFigr.io> service. Supports both 'knitr' and interactive execution within 'RStudio'.

**License** MIT + file LICENSE

**Encoding** UTF-8

**URL** <https://github.com/GoFigr/gofigr>

**BugReports** <https://github.com/GoFigr/gofigr/issues>

**RoxygenNote** 7.3.2

**Imports** httr, jsonlite, knitr, base64enc, readr, magick, qrcode, getPass, textutils, scriptName, xfun, rlang, stringr, rstudioapi, rsvg, cowplot, gridGraphics

**Suggests** ggplot2, gplots

**Config/pak/sysreqs** libmagick++-dev gsfontr libicu-dev librsvg2-dev libssl-dev libx11-dev

**Repository** <https://gofigr.r-universe.dev>

**RemoteUrl** <https://github.com/gofigr/gofigr>

**RemoteRef** HEAD

**RemoteSha** c186220cd888bc0e5ea499ea47f38c2399e64db1

## Contents

authenticate_jwt . . . . .	3
capture . . . . .	3
CONFIG_PATH . . . . .	4
create_analysis . . . . .	4
create_api_key . . . . .	5

create_figure . . . . .	5
create_revision . . . . .	6
create_workspace . . . . .	6
DATA_TYPES . . . . .	7
default_if_null . . . . .	7
enable . . . . .	8
encode_raw_data . . . . .	9
find_analysis . . . . .	9
find_figure . . . . .	10
find_workspace . . . . .	10
get_analysis . . . . .	11
get_api_id . . . . .	11
get_figure . . . . .	12
get_options . . . . .	12
get_revision . . . . .	13
get_revision_url . . . . .	13
get_workspace . . . . .	14
gfconfig . . . . .	14
gf_plot . . . . .	15
gf_print . . . . .	15
gofigr_cat . . . . .	16
gofigr_client . . . . .	16
gofigr_DELETE . . . . .	17
gofigr_GET . . . . .	18
gofigr_make_handler . . . . .	18
gofigr_PATCH . . . . .	19
gofigr_POST . . . . .	19
gofigr_PUT . . . . .	20
infer_workspace . . . . .	20
intercept . . . . .	21
is_expired_token . . . . .	22
LINK_WATERMARK . . . . .	22
list_analyses . . . . .	23
list_workspaces . . . . .	23
login_with_api_key . . . . .	24
login_with_username . . . . .	24
make_code_data . . . . .	25
make_file_data . . . . .	25
make_image_data . . . . .	26
make_raw_data . . . . .	27
make_table_data . . . . .	27
make_text_data . . . . .	28
NO_WATERMARK . . . . .	28
print.gofigr . . . . .	29
print.gofigrdata . . . . .	29
QR_WATERMARK . . . . .	30
read_config . . . . .	30
read_prompt . . . . .	31

<i>authenticate_jwt</i>	3
refresh_jwt . . . . .	31
response_to_JSON . . . . .	32
set_options . . . . .	32
stack_horizontally . . . . .	33
stack_vertically . . . . .	33
suppress . . . . .	34
update_revision_data . . . . .	34
user_info . . . . .	35
watermark_generator . . . . .	35

**Index** **37**

<code>authenticate_jwt</code>	<i>Performs JWT authentication with username and password. Saves tokens in the GoFigr client.</i>
-------------------------------	---------------------------------------------------------------------------------------------------

**Description**

Performs JWT authentication with username and password. Saves tokens in the GoFigr client.

**Usage**

```
authenticate_jwt(gf)
```

**Arguments**

<code>gf</code>	GoFigr client
-----------------	---------------

**Value**

NA

<code>capture</code>	<i>Captures output of an expression and publishes it to GoFigr. This function is helpful if you have multiple function calls which incrementally build a single figure.</i>
----------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Description**

Captures output of an expression and publishes it to GoFigr. This function is helpful if you have multiple function calls which incrementally build a single figure.

**Usage**

```
capture(expr, name = NULL, data = NULL, env = parent.frame())
```

**Arguments**

expr	expression which generates your figure
name	figure name. Use NULL to infer automatically
data	input data to publish with the figure
env	environment in which to evaluate the captured expression (default=parent.frame())

**Value**

result of evaluating your expression

---

CONFIG_PATH	<i>Default path for the config file</i>
-------------	-----------------------------------------

---

**Description**

Default path for the config file

**Usage**

CONFIG\_PATH

**Format**

An object of class character of length 1.

---

create_analysis	<i>Creates a new analysis</i>
-----------------	-------------------------------

---

**Description**

Creates a new analysis

**Usage**

```
create_analysis(gf, name, description = NULL, workspace = NULL)
```

**Arguments**

gf	GoFigr client
name	analysis name
description	analysis description
workspace	analysis will be created under this workspace. Can be a workspace object or an API ID.

**Value**

created analysis

---

create_api_key	<i>Creates a new API key. This function will only succeed if using password authentication.</i>
----------------	-------------------------------------------------------------------------------------------------

---

**Description**

Creates a new API key. This function will only succeed if using password authentication.

**Usage**

```
create_api_key(gf, name)
```

**Arguments**

gf	GoFigr client. Must be using password authentication.
name	human-readable name of the API key to create, e.g. "John's laptop"

**Value**

response JSON. The "token" property will contain the API key if successful.

---

create_figure	<i>Creates a new figure. The created figure will be blank and won't contain any revisions.</i>
---------------	------------------------------------------------------------------------------------------------

---

**Description**

Creates a new figure. The created figure will be blank and won't contain any revisions.

**Usage**

```
create_figure(gf, analysis, name, description = NULL)
```

**Arguments**

gf	GoFigr object
analysis	analysis under which to create the figure
name	name for the new figure
description	description for the new figure

**Value**

created figure object

---

create_revision	<i>Creates a new revision</i>
-----------------	-------------------------------

---

**Description**

Creates a new revision

**Usage**

```
create_revision(gf, figure, metadata = list(), data = NULL)
```

**Arguments**

gf	GoFigr client
figure	figure under which to create the revision
metadata	metadata for the revision, as named list
data	list of Data objects

**Value**

created revision object

---

create_workspace	<i>Creates a new workspace</i>
------------------	--------------------------------

---

**Description**

Creates a new workspace

**Usage**

```
create_workspace(gf, name, description = NULL)
```

**Arguments**

gf	GoFigr client
name	workspace name
description	workspace description

**Value**

created workspace object

---

`DATA_TYPES`*List of data types supported by GoFigr*

---

**Description**

List of data types supported by GoFigr

**Usage**

```
DATA_TYPES
```

**Format**

An object of class `list` of length 5.

**Value**

list where names are human-readable names of data types, and values are corresponding API types (strings).

---

`default_if_null`*Returns a default value if argument is null or empty*

---

**Description**

Returns a default value if argument is null or empty

**Usage**

```
default_if_null(x, default)
```

**Arguments**

<code>x</code>	argument
<code>default</code>	default value if <code>x</code> is null, NA or ""

**Value**

`x` if not null, NA or "", or the default value

---

enable	<i>Enables GoFigr support.</i>
--------	--------------------------------

---

**Description**

Enables GoFigr support.

**Usage**

```
enable(
  analysis_api_id = NULL,
  analysis_name = NULL,
  workspace = NULL,
  create_analysis = TRUE,
  analysis_description = NULL,
  watermark = QR_WATERMARK,
  auto_publish = TRUE,
  verbose = FALSE,
  debug = FALSE,
  show = "watermark"
)
```

**Arguments**

analysis_api_id	Analysis API ID (if analysis_name is NULL)
analysis_name	Analysis name (if analysis_api_id is NULL)
workspace	API ID of the workspace
create_analysis	if TRUE and analysis_name does not exist, it will be automatically created
analysis_description	analysis description if creating a new analysis
watermark	watermark class to use, e.g. QR_WATERMARK, LINK_WATERMARK or NO_WATERMARK
auto_publish	will publish all plots automatically if TRUE. Note that setting this option will re-assign plot() and print() in the global environment.
verbose	whether to show verbose output
debug	whether to show debugging information
show	which figure to display in the document: original, watermark, or hide. Note that this setting \ only affects the display and doesn't change what gets published: e.g. even if you choose to display \ the original figure, the watermarked version will still be published to GoFigr.

**Value**

named list of GoFigr options



---

encode_raw_data	<i>Converts a GoFigr data object into R primitives that can be converted to JSON, performing base64 encoding of binary data.</i>
-----------------	----------------------------------------------------------------------------------------------------------------------------------

---

**Description**

Converts a GoFigr data object into R primitives that can be converted to JSON, performing base64 encoding of binary data.

**Usage**

```
encode_raw_data(data)
```

**Arguments**

data	GoFigr data object
------	--------------------

**Value**

encoded data object

**Examples**

```
data <- make_raw_data("test", "text", list(a=1), charToRaw("abcdefksjdfklsd"))
encode_raw_data(data)
```

---

find_analysis	<i>Finds an analysis by name, optionally creating one if it doesn't exist.</i>
---------------	--------------------------------------------------------------------------------

---

**Description**

Finds an analysis by name, optionally creating one if it doesn't exist.

**Usage**

```
find_analysis(gf, name, description = NULL, workspace = NULL, create = FALSE)
```

**Arguments**

gf	GoFigr client
name	name of the analysis to find
description	description of the analysis if it needs to be created
workspace	parent workspace
create	if TRUE and the analysis doesn't exist, it will be created; throws an error otherwise.

**Value**

analysis object

---

find_figure	<i>Finds a figure by name</i>
-------------	-------------------------------

---

**Description**

Finds a figure by name

**Usage**

```
find_figure(gf, analysis, name, description = NULL, create = FALSE)
```

**Arguments**

gf	GoFigr client
analysis	parent analysis
name	name of the figure to find
description	description of the figure if it needs to be created
create	if TRUE and the figure doesn't exist, it will be created; throws an error otherwise.

**Value**

figure object

---

find_workspace	<i>Finds a workspace by name, optionally creating it if doesn't exist</i>
----------------	---------------------------------------------------------------------------

---

**Description**

Finds a workspace by name, optionally creating it if doesn't exist

**Usage**

```
find_workspace(gf, name, description = NULL, create = FALSE)
```

**Arguments**

gf	GoFigr client
name	name of the workspace to find
description	workspace description if creating a new one
create	if TRUE and workspace is not found, it will be created

**Value**

workspace if found; throws an error if not.

---

get\_analysis                      *Fetches an analysis given an API ID.*

---

**Description**

Fetches an analysis given an API ID.

**Usage**

```
get_analysis(gf, api_id)
```

**Arguments**

gf	GoFigr client
api_id	API ID for the analysis

**Value**

analysis object

---

get\_api\_id                      *Returns obj\$api\_id if argument is an object, or identity if it's a string.*

---

**Description**

Returns obj\$api\_id if argument is an object, or identity if it's a string.

**Usage**

```
get_api_id(obj)
```

**Arguments**

obj	object for which to get the API ID
-----	------------------------------------

**Value**

API ID, a string

---

get_figure	<i>Fetches a figure given an API ID.</i>
------------	------------------------------------------

---

**Description**

Fetches a figure given an API ID.

**Usage**

```
get_figure(gf, api_id)
```

**Arguments**

gf	GoFigr client
api_id	API ID for the figure

**Value**

figure object

---

get_options	<i>Gets configured GoFigr options.</i>
-------------	----------------------------------------

---

**Description**

Gets configured GoFigr options.

**Usage**

```
get_options()
```

**Value**

GoFigr options, or NULL if not set.

---

get_revision	<i>Fetches a revision given an API ID.</i>
--------------	--------------------------------------------

---

**Description**

Fetches a revision given an API ID.

**Usage**

```
get_revision(gf, api_id)
```

**Arguments**

gf	GoFigr client
api_id	API ID for the revision

**Value**

revision object

---

get_revision_url	<i>Gets the full URL for a revision</i>
------------------	-----------------------------------------

---

**Description**

Gets the full URL for a revision

**Usage**

```
get_revision_url(rev)
```

**Arguments**

rev	revision object
-----	-----------------

**Value**

URL, a string

get\_workspace            *Retrieves workspace details.*

---

**Description**

Retrieves workspace details.

**Usage**

```
get_workspace(gf, api_id)
```

**Arguments**

gf	GoFigr client
api_id	API ID of the workspace

**Value**

workspace details

---

gfconfig            *Configures gofigr for use on this machine. Saves configuration to ~/.gofigr.*

---

**Description**

Configures gofigr for use on this machine. Saves configuration to ~/.gofigr.

**Usage**

```
gfconfig(max_attempts = 3)
```

**Arguments**

max_attempts	Maximum number of password attempts
--------------	-------------------------------------

**Value**

No return value

---

gf_plot	<i>Plots and publishes an object (if supported)</i>
---------	-----------------------------------------------------

---

**Description**

Plots and publishes an object (if supported)

**Usage**

```
gf_plot(...)
```

**Arguments**

... passed directly to plot

**Value**

result of the call to plot(...)

---

gf_print	<i>Prints and publishes an object (if supported)</i>
----------	------------------------------------------------------

---

**Description**

Prints and publishes an object (if supported)

**Usage**

```
gf_print(...)
```

**Arguments**

... passed directly to print

**Value**

result of the call to print(...)

---

gofigr_cat	<i>Equivalent to cat but only outputs if GoFigr client is verbose.</i>
------------	------------------------------------------------------------------------

---

**Description**

Equivalent to cat but only outputs if GoFigr client is verbose.

**Usage**

```
gofigr_cat(gf, content, ...)
```

**Arguments**

gf	GoFigr client
content	text to print
...	passed to cat

**Value**

NA

---

gofigr_client	<i>Creates and configures a GoFigr client. You can login either using a username &amp; password or an API key. See examples.</i>
---------------	----------------------------------------------------------------------------------------------------------------------------------

---

**Description**

Username, password, API key and workspace are read from the GoFigr configuration file (~/.gofigr) or environment variables if not supplied:

**Usage**

```
gofigr_client(
  username = NULL,
  password = NULL,
  api_key = NULL,
  url = NULL,
  anonymous = FALSE,
  verbose = FALSE,
  workspace = NULL,
  ignore_config = FALSE
)
```



**Arguments**

username	username (if not using API key)
password	password (if not using API key)
api_key	API key (if not using password authentication)
url	API URL (optional, you generally won't want to modify this)
anonymous	whether to login anonymously
verbose	set to TRUE to enable verbose output
workspace	default workspace (API ID)
ignore_config	if TRUE, will ignore environment variables and other external configuration

**Details**

\* GF\_USERNAME or config\$username \* GF\_PASSWORD or config\$password \* GF\_API\_KEY or config\$api\_key \* GF\_WORKSPACE or config\$workspace \* GF\_URL or config\$url

**Value**

configured GoFigr client which you can pass to other functions

**Examples**

```
## Not run: gofigr_client() # use config from ~/.gofigr or environment variables
## Not run: gofigr_client(username="joe", password="abc123") # password login
## Not run: gofigr_client(api_key="abcdef0123456789") # API key login
```

---

gofigr\_DELETE                      *Wrapper for httr::DELETE that automatically handles authentication.*

---

**Description**

Wrapper for httr::DELETE that automatically handles authentication.

**Usage**

```
gofigr_DELETE(gf, url, expected_status_code = 200, ...)
```

**Arguments**

gf	configured GoFigr client
url	URL to make the request to, relative to the API URL e.g. user/
expected_status_code	expected HTTP response code. We will throw an exception if it differs.
...	passed to the httr request function

**Value**

result of calling the underlying httr request function

---

gofigr_GET	<i>Wrapper for httr::GET that automatically handles authentication.</i>
------------	-------------------------------------------------------------------------

---

**Description**

Wrapper for httr::GET that automatically handles authentication.

**Usage**

```
gofigr_GET(gf, url, expected_status_code = 200, ...)
```

**Arguments**

gf	configured GoFigr client
url	URL to make the request to, relative to the API URL e.g. user/
expected_status_code	expected HTTP response code. We will throw an exception if it differs.
...	passed to the httr request function

**Value**

result of calling the underlying httr request function

---

gofigr_make_handler	<i>Wraps an HTTR method e.g. GET to provide relative URL resolution and authentication</i>
---------------------	--------------------------------------------------------------------------------------------

---

**Description**

Wraps an HTTR method e.g. GET to provide relative URL resolution and authentication

**Usage**

```
gofigr_make_handler(name, method)
```

**Arguments**

name	method name, e.g. "GET"
method	HTTR method, e.g. httr::GET

**Value**

wrapped method which takes a GoFigr client, a relative URL and an expected HTTP status code.

---

gofigr_PATCH	<i>Wrapper for http::PATCH that automatically handles authentication.</i>
--------------	---------------------------------------------------------------------------

---

**Description**

Wrapper for http::PATCH that automatically handles authentication.

**Usage**

```
gofigr_PATCH(gf, url, expected_status_code = 200, ...)
```

**Arguments**

gf	configured GoFigr client
url	URL to make the request to, relative to the API URL e.g. user/
expected_status_code	expected HTTP response code. We will throw an exception if it differs.
...	passed to the http request function

**Value**

result of calling the underlying http request function

---

gofigr_POST	<i>Wrapper for http::POST that automatically handles authentication.</i>
-------------	--------------------------------------------------------------------------

---

**Description**

Wrapper for http::POST that automatically handles authentication.

**Usage**

```
gofigr_POST(gf, url, expected_status_code = 200, ...)
```

**Arguments**

gf	configured GoFigr client
url	URL to make the request to, relative to the API URL e.g. user/
expected_status_code	expected HTTP response code. We will throw an exception if it differs.
...	passed to the http request function

**Value**

result of calling the underlying http request function

---

gofigr_PUT	<i>Wrapper for htrr::PUT that automatically handles authentication.</i>
------------	-------------------------------------------------------------------------

---

**Description**

Wrapper for htrr::PUT that automatically handles authentication.

**Usage**

```
gofigr_PUT(gf, url, expected_status_code = 200, ...)
```

**Arguments**

gf	configured GoFigr client
url	URL to make the request to, relative to the API URL e.g. user/
expected_status_code	expected HTTP response code. We will throw an exception if it differs.
...	passed to the htrr request function

**Value**

result of calling the underlying htrr request function

---

infer_workspace	<i>Returns the argument if a valid workspace is passed, or the default workspace from the GoFigr client otherwise. Throws an error if both are NULL.</i>
-----------------	----------------------------------------------------------------------------------------------------------------------------------------------------------

---

**Description**

Returns the argument if a valid workspace is passed, or the default workspace from the GoFigr client otherwise. Throws an error if both are NULL.

**Usage**

```
infer_workspace(gf, workspace)
```

**Arguments**

gf	GoFigr client
workspace	workspace or NULL

**Value**

workspace object

---

intercept	<i>Wraps a plotting function (e.g. heatmap.2) so that its output is intercepted by GoFigr.</i>
-----------	------------------------------------------------------------------------------------------------

---

### Description

Wraps a plotting function (e.g. heatmap.2) so that its output is intercepted by GoFigr.

### Usage

```
intercept(  
  plot_func,  
  supported_classes = NULL,  
  force = FALSE,  
  data = NULL,  
  figure_name = NULL  
)
```

### Arguments

plot_func	function to intercept
supported_classes	calls will be intercepted *only if* the first argument is an \ instance of any of these classes
force	force intercept even if auto_publish is off
data	data to publish with the figure
figure_name	name of the figure, if different from default

### Value

intercepted function

### Examples

```
heatmap.2 <- intercept(gplots::heatmap.2)
```

is\_expired\_token      *Returns True if the response indicates an expired JWT token*

---

**Description**

Returns True if the response indicates an expired JWT token

**Usage**

```
is_expired_token(res)
```

**Arguments**

res                    htrr response

**Value**

True if token expired

---

LINK\_WATERMARK      *Draws a watermark with just a GoFigr link*

---

**Description**

Draws a watermark with just a GoFigr link

**Usage**

```
LINK_WATERMARK(revision, image)
```

**Arguments**

revision              GoFigr revision object for which to generate a watermark  
image                 Magick image to which to add the watermark

**Value**

a function which you can pass to enable\_knitr(watermark)

---

list_analyses	<i>Lists analyses under a workspace.</i>
---------------	------------------------------------------

---

**Description**

Lists analyses under a workspace.

**Usage**

```
list_analyses(gf, workspace_id = NULL)
```

**Arguments**

gf	GoFigr client
workspace_id	API id of the workspace

**Value**

list of analyses

---

list_workspaces	<i>List all workspaces available to the user.</i>
-----------------	---------------------------------------------------

---

**Description**

List all workspaces available to the user.

**Usage**

```
list_workspaces(gf)
```

**Arguments**

gf	GoFigr client
----	---------------

**Value**

List of workspaces

login\_with\_api\_key     *Prompts the user for an API key or creates a new one*

---

**Description**

Prompts the user for an API key or creates a new one

**Usage**

```
login_with_api_key(gf, max_attempts)
```

**Arguments**

gf	Password-authenticated GoFigr client
max_attempts	Maximum number of login attempts before giving up

**Value**

API key, either supplied by the user or newly created

---

login\_with\_username     *Prompts the user for username & password and logs into GoFigr*

---

**Description**

Prompts the user for username & password and logs into GoFigr

**Usage**

```
login_with_username(max_attempts)
```

**Arguments**

max_attempts	maximum number of login attempts before giving up
--------------	---------------------------------------------------

**Value**

GoFigr client



---

make_code_data	<i>Creates a GoFigr data object storing source code</i>
----------------	---------------------------------------------------------

---

**Description**

Creates a GoFigr data object storing source code

**Usage**

```
make_code_data(  
    name,  
    contents_or_file,  
    language,  
    format = "text",  
    metadata = NULL  
)
```

**Arguments**

name	name of this code object
contents_or_file	contents, a character string or file object
language	programming language, e.g. Python or R
format	not supported at the moment; please use the default
metadata	metadata associated with this object

**Value**

GoFigr data object

---

make_file_data	<i>Creates a GoFigr data object storing file data</i>
----------------	-------------------------------------------------------

---

**Description**

Creates a GoFigr data object storing file data

**Usage**

```
make_file_data(name, file_or_raw, path = NULL, metadata = NULL)
```

**Arguments**

name	name of this file
file_or_raw	image data, either a file or a raw vector
path	file path
metadata	metadata associated with this file

**Value**

GoFigr data object

---

make_image_data	<i>Creates a GoFigr data object storing image data</i>
-----------------	--------------------------------------------------------

---

**Description**

Creates a GoFigr data object storing image data

**Usage**

```
make_image_data(name, file_or_raw, format, is_watermarked, metadata = NULL)
```

**Arguments**

name	name of this image
file_or_raw	image data, either a file or a raw vector
format	format, e.g. "png"
is_watermarked	whether this file has a GoFigr watermark
metadata	metadata associated with this image

**Value**

GoFigr data object

---

make_raw_data	<i>Creates a GoFigr data object which can be attached to revisions.</i>
---------------	-------------------------------------------------------------------------

---

**Description**

Creates a GoFigr data object which can be attached to revisions.

**Usage**

```
make_raw_data(name, type, metadata, data)
```

**Arguments**

name	name of this data
type	data type, e.g. DATA_TYPE\$image
metadata	metadata associated with this data object
data	raw bytes

**Value**

data object

---

make_table_data	<i>Creates a GoFigr data object storing data.frame/tabular data</i>
-----------------	---------------------------------------------------------------------

---

**Description**

Creates a GoFigr data object storing data.frame/tabular data

**Usage**

```
make_table_data(name, frame, metadata = NULL)
```

**Arguments**

name	name of this data object
frame	data.frame
metadata	metadata associated with this data object

**Value**

GoFigr data object

make\_text\_data            *Creates a GoFigr data object to store text*

---

**Description**

Creates a GoFigr data object to store text

**Usage**

```
make_text_data(name, contents, metadata = NULL)
```

**Arguments**

name	name of this data object
contents	contents, a character string
metadata	metadata associated with this object

**Value**

GoFigr data object

---

NO\_WATERMARK            *Does not draw any watermarks.*

---

**Description**

Does not draw any watermarks.

**Usage**

```
NO_WATERMARK
```

**Format**

An object of class NULL of length 0.

**Value**

does not return anything (NULL)

---

print.gofigr	<i>Default print method for a GoFigr client.</i>
--------------	--------------------------------------------------

---

**Description**

Default print method for a GoFigr client.

**Usage**

```
## S3 method for class 'gofigr'  
print(x, ...)
```

**Arguments**

x	GoFigr client
...	passed to cat

**Value**

NA

---

print.gofigrdata	<i>Default print representation of GoFigr data objects.</i>
------------------	-------------------------------------------------------------

---

**Description**

Default print representation of GoFigr data objects.

**Usage**

```
## S3 method for class 'gofigrdata'  
print(x, ...)
```

**Arguments**

x	object to print
...	passed to cat

**Value**

NA

---

QR_WATERMARK	<i>Draws a watermark with a GoFigr link and a QR code</i>
--------------	-----------------------------------------------------------

---

**Description**

Draws a watermark with a GoFigr link and a QR code

**Usage**

```
QR_WATERMARK(revision, image)
```

**Arguments**

revision	GoFigr revision object for which to generate a watermark
image	Magick image to which to add the watermark

**Value**

a function which you can pass to `enable_knitr(watermark)`

---

read_config	<i>Reads the GoFigr configuration, prioritizing environment variables over the config file:</i>
-------------	-------------------------------------------------------------------------------------------------

---

**Description**

\* GF\_USERNAME or config["username"] \* GF\_PASSWORD or config["password"] \* GF\_API\_KEY or config["api\_key"] \* GF\_WORKSPACE or config["workspace"] \* GF\_URL or config["url"]

**Usage**

```
read_config(path = CONFIG_PATH)
```

**Arguments**

path	path to the config file, default <code>~/gofigr</code>
------	--------------------------------------------------------

**Value**

parsed configuration or empty list if not available

---

read_prompt	<i>Reads a prompt from stdin and performs optional validation</i>
-------------	-------------------------------------------------------------------

---

**Description**

Reads a prompt from stdin and performs optional validation

**Usage**

```
read_prompt(prompt, validate = NULL, attempt = 1, max_attempts = 2)
```

**Arguments**

prompt	prompt, e.g. "Enter username: "
validate	function input => transformed input if input is valid or, error
attempt	current attempt at getting a valid input
max_attempts	maximum number of attempts

**Value**

input, or result of validate(input) if validate is supplied

---

refresh_jwt	<i>Refreshes the JWT access token. Attempts re-authentication if refresh fails.</i>
-------------	-------------------------------------------------------------------------------------

---

**Description**

Refreshes the JWT access token. Attempts re-authentication if refresh fails.

**Usage**

```
refresh_jwt(gf)
```

**Arguments**

gf	GoFigr client.
----	----------------

**Value**

NA

---

response_to_JSON	<i>Convenience function for parsing JSON from htr responses</i>
------------------	-----------------------------------------------------------------

---

**Description**

Convenience function for parsing JSON from htr responses

**Usage**

```
response_to_JSON(response)
```

**Arguments**

response	htr response
----------	--------------

**Value**

parsed JSON

---

set_options	<i>Sets GoFigr options.</i>
-------------	-----------------------------

---

**Description**

Sets GoFigr options.

**Usage**

```
set_options(options)
```

**Arguments**

options	New options that will replace existing options.
---------	-------------------------------------------------

**Value**

NA



---

`stack_horizontally`     *Stacks images horizontally, centering them vertically.*

---

**Description**

Stacks images horizontally, centering them vertically.

**Usage**

```
stack_horizontally(images)
```

**Arguments**

`images`                vector of images to stack

**Value**

composite image

---

`stack_vertically`     *Stacks images vertically, centering them horizontally.*

---

**Description**

Stacks images vertically, centering them horizontally.

**Usage**

```
stack_vertically(images)
```

**Arguments**

`images`                vector of images to stack

**Value**

composite image

---

suppress	<i>Suppresses any automatic GoFigr publication hooks.</i>
----------	-----------------------------------------------------------

---

**Description**

Suppresses any automatic GoFigr publication hooks.

**Usage**

```
suppress(func)
```

**Arguments**

func	function in which to suppress intercepts
------	------------------------------------------

**Value**

the function with GoFigr suppressed

---

update_revision_data	<i>Updates data associated with a figure</i>
----------------------	----------------------------------------------

---

**Description**

Updates data associated with a figure

**Usage**

```
update_revision_data(gf, revision, new_data, silent = FALSE)
```

**Arguments**

gf	GoFigr client
revision	revision or its API ID for which to update the data
new_data	new data, as a list of GoFigrData objects (e.g. make_image_data or make_text_data)
silent	whether to generate an activity. Internal use only.

**Value**

updated revision

---

user_info	<i>Fetches user details for the currently logged in user.</i>
-----------	---------------------------------------------------------------

---

**Description**

Fetches user details for the currently logged in user.

**Usage**

```
user_info(gf)
```

**Arguments**

gf	GoFigr client
----	---------------

**Value**

user details

---

watermark_generator	<i>Makes a watermark generator. You can use the result with enable(watermark=...).</i>
---------------------	----------------------------------------------------------------------------------------

---

**Description**

Makes a watermark generator. You can use the result with enable(watermark=...).

**Usage**

```
watermark_generator(  
  show_qr = TRUE,  
  qr_size_px = c(100, 100),  
  link_size_px = c(500, 100),  
  link_bg = "#ffffff",  
  font_color = "#000000",  
  font_size = 14,  
  font = "mono",  
  dynamic_size = TRUE  
)
```

**Arguments**

<code>show_qr</code>	show QR code
<code>qr_size_px</code>	two-element vector specifying the width, height of the QR code
<code>link_size_px</code>	two-element vector specifying the width, height of the link
<code>link_bg</code>	background color for the link
<code>font_color</code>	font color for the link
<code>font_size</code>	font size for the link
<code>font</code>	font name or family, e.g. "mono"
<code>dynamic_size</code>	whether to automatically adjust the watermark size depending on the size of the current graphics device

**Value**

a function which you can pass to `enable_knitr(watermark)`

# Index

## \* datasets

- CONFIG\_PATH, 4
- DATA\_TYPES, 7
- NO\_WATERMARK, 28

authenticate\_jwt, 3

capture, 3  
CONFIG\_PATH, 4  
create\_analysis, 4  
create\_api\_key, 5  
create\_figure, 5  
create\_revision, 6  
create\_workspace, 6

DATA\_TYPES, 7  
default\_if\_null, 7

enable, 8  
encode\_raw\_data, 9

find\_analysis, 9  
find\_figure, 10  
find\_workspace, 10

get\_analysis, 11  
get\_api\_id, 11  
get\_figure, 12  
get\_options, 12  
get\_revision, 13  
get\_revision\_url, 13  
get\_workspace, 14  
gf\_plot, 15  
gf\_print, 15  
gfconfig, 14  
gofigr\_cat, 16  
gofigr\_client, 16  
gofigr\_DELETE, 17  
gofigr\_GET, 18  
gofigr\_make\_handler, 18  
gofigr\_PATCH, 19

gofigr\_POST, 19  
gofigr\_PUT, 20  
  
infer\_workspace, 20  
intercept, 21  
is\_expired\_token, 22

LINK\_WATERMARK, 22  
list\_analyses, 23  
list\_workspaces, 23  
login\_with\_api\_key, 24  
login\_with\_username, 24

make\_code\_data, 25  
make\_file\_data, 25  
make\_image\_data, 26  
make\_raw\_data, 27  
make\_table\_data, 27  
make\_text\_data, 28

NO\_WATERMARK, 28  
  
print.gofigr, 29  
print.gofigrdata, 29

QR\_WATERMARK, 30  
  
read\_config, 30  
read\_prompt, 31  
refresh\_jwt, 31  
response\_to\_JSON, 32

set\_options, 32  
stack\_horizontally, 33  
stack\_vertically, 33  
suppress, 34

update\_revision\_data, 34  
user\_info, 35

watermark\_generator, 35