Package: httpgd (via r-universe)

October 1, 2026

Type Package

Title A 'HTTP' Server Graphics Device

Version 2.1.4

Description A graphics device for R that is accessible via network protocols. This package was created to make it easier to embed live R graphics in integrated development environments and other applications. The included 'HTML/JavaScript' client (plot viewer) aims to provide a better overall user experience when dealing with R graphics. The device asynchronously serves graphics via 'HTTP' and 'WebSockets'.

License GPL (>= 2)

Depends R (>= 4.2.0)

Imports unigd (>= 0.2.0)

LinkingTo unigd, cpp11 (>= 0.2.4), AsioHeaders (>= 1.28.2)

Suggests testthat, xml2 (>= 1.0.0), knitr, rmarkdown, covr, future, httr, jsonlite

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Encoding UTF-8

URL https://github.com/nx10/httpgd, https://nx10.dev/httpgd/

BugReports https://github.com/nx10/httpgd/issues

VignetteBuilder knitr

Config/pak/sysreqs libcairo2-dev libfontconfig1-dev libfreetype6-dev libpng-dev libtiff-dev zlib1g-dev

Repository https://nx10.r-universe.dev

Date/Publication 2026-02-27 17:39:15 UTC

RemoteUrl https://github.com/nx10/httpgd

RemoteRef HEAD

RemoteSha 087ebf5bbf46ddc52e107d6a8bb26381177d2e7f

Contents

httpgd-package . . . 2
hgd . . . 3
hgd_browse . . . 4
hgd_close . . . 5
hgd_details . . . 6
hgd_generate_token . . . 7
hgd_url . . . 7
hgd_view . . . 8
hgd_watch . . . 9

Index **11**

`httpgd-package` *httpgd: HTTP server graphics device*

Description

Asynchronous HTTP server graphics device.

Author(s)

Maintainer: Florian Rupprecht <floruppr@gmail.com> (ORCID)

Other contributors:

- Kun Ren <mail@renkun.me> [contributor]
- Jeroen Ooms <jeroen@berkeley.edu> (ORCID) [contributor]

See Also

Useful links:

- https://github.com/nx10/httpgd
- https://nx10.dev/httpgd/
- Report bugs at https://github.com/nx10/httpgd/issues

hgd *Asynchronous HTTP server graphics device.*

Description

This function initializes a httpgd graphics device and starts a local webserver, that allows for access via HTTP and WebSockets. A link will be printed by which the web client can be accessed using a browser.

Usage

```
hgd(
  host = getOption("httpgd.host", "127.0.0.1"),
  port = getOption("httpgd.port", 0),
  cors = getOption("httpgd.cors", FALSE),
  token = getOption("httpgd.token", TRUE),
  silent = getOption("httpgd.silent", FALSE),
  width = getOption("httpgd.width", 720),
  height = getOption("httpgd.height", 576),
  zoom = getOption("httpgd.zoom", 1),
  bg = getOption("httpgd.bg", "white"),
  pointsize = getOption("httpgd.pointsize", 12),
  system_fonts = getOption("httpgd.system_fonts", list()),
  user_fonts = getOption("httpgd.user_fonts", list()),
  reset_par = getOption("httpgd.reset_par", FALSE)
)
```

Arguments

host	Server hostname. Set to `"0.0.0.0"` to enable remote access. We recommend to **only enable remote access in trusted networks**. The network security of httpgd has not yet been properly tested.
port	Server port. If this is set to `0`, an open port will be assigned.
cors	Toggles Cross-Origin Resource Sharing (CORS) header. When set to `TRUE`, CORS header will be set to `"*"`.
token	(Optional) security token. When set, all requests need to include a token to be allowed. (Either in a request header (`X-HTTPGD-TOKEN`) field or as a query parameter.) This parameter can be set to `TRUE` to generate a random 8 character alphanumeric token. A random token of the specified length is generated when it is set to a number. `FALSE` deactivates the token.
silent	When set to `FALSE` no information will be printed to console.
width	Initial plot width (pixels).
height	Initial plot height (pixels).
zoom	Initial plot zoom level (For example: 2 corresponds to 200%, 0.5 would be 50%.).

bg	Background color.
pointsize	Graphics device point size.
system_fonts	Named list of font names to be aliased with fonts installed on your system. If unspecified, the R default families sans, serif, mono and symbol are aliased to the family returned by systemfonts::font_info().
user_fonts	Named list of fonts to be aliased with font files provided by the user rather than fonts properly installed on the system. The aliases can be fonts from the fontquiver package, strings containing a path to a font file, or a list containing name and file elements with name indicating the font alias in the SVG output and file the path to a font file.
reset_par	If set to TRUE, global graphics parameters will be saved on device start and reset every time the plots are cleared (see graphics::par()).

Details

All font settings and descriptions are adopted from the excellent 'svglite' package.

Value

No return value, called to initialize graphics device.

Examples

```
## Not run:

hgd() # Initialize graphics device and start server
hgd_browse() # Or copy the displayed link in the browser

# Plot something
x <- seq(0, 3 * pi, by = 0.1)
plot(x, sin(x), type = "l")

dev.off() # alternatively: hgd_close()

## End(Not run)
```

---

hgd_browse *Open httpgd URL in the browser.*

Description

This function will only work after starting a device with hgd().

Usage

```
hgd_browse(..., which = dev.cur(), browser = getOption("browser"))
```

Arguments

...	Parameters passed to hgd_url().
which	Which device (ID).
browser	Program to be used as HTML browser.

Value

URL.

Examples

```
## Not run:

hgd()
hgd_browse() # open browser
hist(rnorm(100))

dev.off()

## End(Not run)
```

---

hgd_close *Close httpgd device.*

Description

This achieves the same effect as grDevices::dev.off(), but will only close the device if it has the httpgd type.

Usage

```
hgd_close(which = dev.cur(), all = FALSE)
```

Arguments

which	Which device (ID).
all	Should all running httpgd devices be closed.

Value

Number and name of the new active device (after the specified device has been shut down).

Examples

```
## Not run:

hgd()
hgd_browse() # open browser
hist(rnorm(100))
hgd_close()

hgd()
hgd()
hgd()
hgd_close(all = TRUE)

## End(Not run)
```

`hgd_details` *httpgd device status.*

Description

Access status information of a httpgd graphics device. This function will only work after starting a device with `hgd()`.

Usage

```
hgd_details(which = dev.cur())
```

Arguments

`which`	Which device (ID).

Value

List of status variables with the following named items: `$host`: Server hostname, `$port`: Server port, `$token`: Security token, `$hsize`: Plot history size (how many plots are accessible), `$upid`: Update ID (changes when the device has received new information), `$active`: Is the device the currently activated device.

Examples

```
## Not run:

hgd()
hgd_details()
plot(1, 1)
hgd_details()

dev.off()

## End(Not run)
```

hgd_generate_token *Generate random alphanumeric token.*

Description

This is mainly used internally by httpgd, but exposed for testing purposes.

Usage

```
hgd_generate_token(len)
```

Arguments

`len`	Token length (number of characters).

Value

Random token string.

Examples

```
hgd_generate_token(6)
```

hgd_url *httpgd URL.*

Description

Generate URLs to the plot viewer or to plot SVGs. This function will only work after starting a device with `hgd()`.

Usage

```
hgd_url(
  endpoint = "live",
  which = dev.cur(),
  host = NA,
  port = NA,
  explicit = FALSE,
  omit_token = FALSE,
  ...
)
```

Arguments

endpoint	API endpoint. The default, "live" is the HTML/JS plot viewer. Can be set to a numeric plot index or plot ID (see unigd::ugd_id()) to obtain the direct URL to the SVG.
which	Which device (ID).
host	Replaces hostname.
port	Replaces port.
explicit	Ads hgd={host}:{port} query parameter. Needed for host resolution in some editors.
omit_token	Should the security token be omitted from the URL.
...	Other query parameters that will be appended to the URL.

Details

Note: If the included client is used set websockets=0 or sidebar=0 to turn off WebSocket or plot history sidebar.

Value

URL.

Examples

```
## Not run:

hgd()
my_url <- hgd_url()
hgd_url(0)
hgd_url(plot_id(), width = 800, height = 600)

dev.off()

## End(Not run)
```

---

hgd_view *Open httpgd URL in the IDE.*

Description

Global option viewer needs to be set to a function that accepts the client URL as a parameter.

Usage

```
hgd_view()
```

Details

This function will only work after starting a device with `hgd()`.

Value

`viewer` function return value.

Examples

```
## Not run:

hgd()
hgd_view()
hist(rnorm(100))

dev.off()

## End(Not run)
```

`hgd_watch` *Watch for changes in code files and refresh plots automatically.*

Description

This function may be used to rapidly develop visualizations by replacing a workflow of reloading and executing code manually.

Usage

```
hgd_watch(
  watch = list.files(pattern = "\\.R$", ignore.case = TRUE),
  on_change = function(changed_files) {
     print(changed_files)
 },
  interval = 1,
  on_start = hgd_browse,
  on_exit = NULL,
  on_error = print,
  reset_par = TRUE,
  ...
)
```

Arguments

`watch`	Paths that are watched for changes (see `utils::fileSnapshot()`)
`on_change`	Will be called when a file changes.
`interval`	Time interval in which changes are detected (in seconds).

on_start	Will be called after the httpgd server is started (may be set to NULL).
on_exit	Will be called before the server is closed (may be set to NULL).
on_error	Will be called when on_change throws an error (may be set to NULL).
reset_par	If set to TRUE, global graphics parameters will be saved on device start and reset every time unigd::ugd_clear() is called (see graphics::par()).
...	Additional parameters passed to hgd(webserver=FALSE, ...)

Value

No return value, called for side effects.

Examples

```
## Not run:

# --- my_code.R ---

plot(rnorm(100), col = "red")

# --- Other file / interactive ---

hgd_watch(
  watch = "my_code.R", # When my_code.R changes...
  on_change = function(...) {
    source("my_code.R") # ...call updated plot function.
  }
)

## End(Not run)
```

Index

graphics::par(), *4*, *10*
grDevices::dev.off(), *5*

hgd, 3
hgd(), *4*, *6*, *7*, *9*
hgd_browse, 4
hgd_close, 5
hgd_details, 6
hgd_generate_token, 7
hgd_url, 7
hgd_url(), *5*
hgd_view, 8
hgd_watch, 9
httpgd *(*httpgd-package*)*, 2
httpgd-package, 2

unigd::ugd_clear(), *10*
unigd::ugd_id(), *8*
utils::fileSnapshot(), *9*