

# Package ‘splashr’

August 29, 2017

**Type** Package

**Title** Tools to Work with the 'Splash' 'JavaScript' Rendering and Scraping Service

**Version** 0.4.0

**Date** 2017-08-26

**Encoding** UTF-8

**Maintainer** Bob Rudis <bob@rud.is>

**Description** 'Splash' <<https://github.com/scrapinghub/splash>> is a 'JavaScript' rendering service. It is a lightweight web browser with an 'HTTP' API, implemented in 'Python' using 'Twisted' and 'QT' and provides some of the core functionality of the 'RSelenium' or 'seleniumPipes' R packages in a lightweight footprint. Some of 'Splash' features include the ability to process multiple web pages in parallel; retrieving 'HTML' results and/or take screen shots; disabling images or use 'Adblock Plus' rules to make rendering faster; executing custom 'JavaScript' in page context; getting detailed rendering info in 'HAR' format.

**URL** <http://github.com/hrbrmstr/splashr>

**BugReports** <https://github.com/hrbrmstr/splashr/issues>

**License** AGPL

**Suggests** testthat, tibble, jpeg, png, covr, knitr, rmarkdown

**Depends** R (>= 3.2.0)

**Imports** xml2, curl, httr, clipr, purrr, stats, utils, docker, magick, scales, formatR, openssl, stringi, jsonlite, HARtools, lubridate

**RoxygenNote** 6.0.1

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Bob Rudis [aut, cre]

**Repository** CRAN

**Date/Publication** 2017-08-29 21:41:48 UTC

**R topics documented:**

|                                    |    |
|------------------------------------|----|
| as_har . . . . .                   | 3  |
| as_htrr_req . . . . .              | 3  |
| as_response . . . . .              | 4  |
| execute_lua . . . . .              | 4  |
| get_content_size . . . . .         | 6  |
| get_content_type . . . . .         | 6  |
| get_har_entry . . . . .            | 7  |
| get_request_type . . . . .         | 8  |
| get_request_url . . . . .          | 8  |
| get_response_body . . . . .        | 9  |
| har_entries . . . . .              | 9  |
| har_entry_count . . . . .          | 10 |
| install_splash . . . . .           | 10 |
| json_fromb64 . . . . .             | 11 |
| killall_splash . . . . .           | 11 |
| render_har . . . . .               | 11 |
| render_html . . . . .              | 13 |
| render_jpeg . . . . .              | 14 |
| render_json . . . . .              | 16 |
| render_png . . . . .               | 18 |
| splash . . . . .                   | 20 |
| splashr . . . . .                  | 20 |
| splashr-exports . . . . .          | 21 |
| splash_active . . . . .            | 21 |
| splash_add_lua . . . . .           | 22 |
| splash_click . . . . .             | 22 |
| splash_enable_javascript . . . . . | 23 |
| splash_focus . . . . .             | 24 |
| splash_go . . . . .                | 24 |
| splash_har . . . . .               | 25 |
| splash_har_reset . . . . .         | 26 |
| splash_history . . . . .           | 26 |
| splash_html . . . . .              | 27 |
| splash_images . . . . .            | 28 |
| splash_perf_stats . . . . .        | 28 |
| splash_plugins . . . . .           | 29 |
| splash_png . . . . .               | 30 |
| splash_press . . . . .             | 30 |
| splash_private_mode . . . . .      | 31 |
| splash_release . . . . .           | 32 |
| splash_response_body . . . . .     | 32 |
| splash_send_keys . . . . .         | 33 |
| splash_send_text . . . . .         | 34 |
| splash_user_agent . . . . .        | 34 |
| splash_version . . . . .           | 36 |
| splash_wait . . . . .              | 36 |

|                                     |           |
|-------------------------------------|-----------|
| <code>as_har</code>                 | 3         |
| <code>start_splash</code> . . . . . | 37        |
| <code>stop_splash</code> . . . . .  | 38        |
| <b>Index</b>                        | <b>39</b> |

---

|                     |   |
|---------------------|---|
| <code>as_har</code> | <i>Turn a generic Splash HAR response into a HAR object</i> |
|---------------------|---|

---

**Description**

Turn a generic Splash HAR response into a HAR object

**Usage**

```
as_har(splash_resp)
```

**Arguments**

`splash_resp`     splash response object

---

|                         |  |
|-------------------------|--|
| <code>as_htr_req</code> | <i>Create an htr verb request function from an HAR request</i> |
|-------------------------|--|

---

**Description**

This function is very useful if you used `splshr` to find XHR requests in a dynamic page and want to be able to make a call directly to that XHR resource. Once you identify the proper HAR entry, pass it to this function and fully working function that makes an `htr::VERB()` request will be created and returned. The text of the function will also be put onto the clipboard if `add_clip` is `TRUE`

**Usage**

```
as_htr_req(entry, quiet = TRUE, add_clip = TRUE)
```

**Arguments**

`entry`             HAR entry  
`quiet`             quiet (no messages)  
`add_clip`          add clip (paste the function text to the clipboard)

---

 as\_response

*Return a HAR entry response as an htr::response object*


---

### Description

Return a HAR entry response as an htr::response object

### Usage

```
as_response(har_entry)
```

### Arguments

har\_entry      a HAR object (should contain a response body to be most useful)

### Examples

```
## Not run:
library(purrr)

URL <- "http://www.svs.cl/portal/principal/605/w3-propertyvalue-18554.html"

splash_local %>%
  splash_response_body(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go(URL) %>%
  splash_wait(2) %>%
  splash_har() -> har

keep(har$log$entries, is_xhr) %>%
  map(as_request) %>%
  map(htr::content, as="parsed")

## End(Not run)
```

---

 execute\_lua

*Execute a custom rendering script and return a result.*


---

### Description

Execute a custom rendering script and return a result.

### Usage

```
execute_lua(splash_obj, lua_source, timeout = 30, allowed_domains, proxy,
  filters, save_args, load_args)
```

**Arguments**

|                 |   |
|-----------------|---|
| splash_obj      | Object created by a call to <a href="#">splash()</a>  |
| lua_source      | Browser automation script. See <a href="#">Splash Script Tutorial</a> for more info.  |
| timeout         | A timeout (in seconds) for the render (defaults to 30). Without reconfiguring the startup parameters of the Splash server (not this package) the maximum allowed value for the timeout is 60 seconds. |
| allowed_domains | Comma-separated list of allowed domain names. If present, Splash won't load anything neither from domains not in this list nor from subdomains of domains not in this list.                           |
| proxy           | Proxy profile name or proxy URL.  |
| filters         | Comma-separated list of request filter names.   |
| save_args       | A list of argument names to put in cache.   |
| load_args       | Parameter values to load from cache   |

**Value**

raw content from the `httr` call. Given the vast diversity of possible return values, it's up to the caller to handle the return value.

**See Also**

Other `splash_renderer`s: [render\\_har](#), [render\\_html](#), [render\\_jpeg](#), [render\\_json](#), [render\\_png](#)

**Examples**

```
## Not run:
splash_local %>%
  execute_lua('
function main(splash)
  splash:go("https://projects.fivethirtyeight.com/congress-trump-score/")
  splash:wait(0.5)
  return splash:evaljs("memberScores")
end
') -> res

rawToChar(res) %>%
  jsonlite::fromJSON(flatten=TRUE) %>%
  purrr::map(tibble::as_tibble) -> member_scores

member_scores

## End(Not run)
```

---

get\_content\_size      *Retrieve size of content | body | headers*

---

**Description**

Retrieve size of content | body | headers

**Usage**

```
get_content_size(har_resp_obj)
```

```
get_body_size(har_resp_obj)
```

```
get_headers_size(har_resp_obj)
```

**Arguments**

har\_resp\_obj      HAR response object

**See Also**

Other splash\_har\_helpers: [get\\_content\\_type](#), [get\\_har\\_entry](#), [get\\_request\\_type](#), [get\\_request\\_url](#), [get\\_response\\_body](#), [har\\_entry\\_count](#)

---

get\_content\_type      *Retrieve or test content type of a HAR request object*

---

**Description**

Retrieve or test content type of a HAR request object

**Usage**

```
get_content_type(har_resp_obj)
```

```
is_content_type(har_resp_obj, type = "application/json")
```

```
is_json(har_resp_obj)
```

```
is_xml(har_resp_obj)
```

```
is_css(har_resp_obj)
```

```
is_plain(har_resp_obj)
```

```
is_binary(har_resp_obj)
```

```
is_javascript(har_resp_obj)
```

```
is_html(har_resp_obj)
```

```
is_jpeg(har_resp_obj)
```

```
is_png(har_resp_obj)
```

```
is_svg(har_resp_obj)
```

```
is_gif(har_resp_obj)
```

```
is_xhr(har_resp_obj)
```

### Arguments

har\_resp\_obj    a reponse object from [render\_har()] or [execute\_lua()]  
type            content type to compare to (default: "application/json")

### See Also

Other splash\_har\_helpers: [get\\_content\\_size](#), [get\\_har\\_entry](#), [get\\_request\\_type](#), [get\\_request\\_url](#), [get\\_response\\_body](#), [har\\_entry\\_count](#)

---

|               |   |
|---------------|---|
| get_har_entry | <i>Retrieve an entry by index from a HAR object</i> |
|---------------|---|

---

### Description

Retrieve an entry by index from a HAR object

### Usage

```
get_har_entry(x, i = 1)
```

### Arguments

x                can be a 'har' object, 'harlog' object or 'harentries' object  
i                index of the HAR entry to retrieve

### See Also

Other splash\_har\_helpers: [get\\_content\\_size](#), [get\\_content\\_type](#), [get\\_request\\_type](#), [get\\_request\\_url](#), [get\\_response\\_body](#), [har\\_entry\\_count](#)

---

|                               |                                      |
|-------------------------------|--------------------------------------|
| <code>get_request_type</code> | <i>Retrieve or test request type</i> |
|-------------------------------|--------------------------------------|

---

**Description**

Retrieve or test request type

**Usage**

```
get_request_type(har_resp_obj)
```

```
is_get(har_resp_obj)
```

```
is_post(har_resp_obj)
```

**Arguments**

`har_resp_obj` HAR response object

**See Also**

Other splash\_har\_helpers: [get\\_content\\_size](#), [get\\_content\\_type](#), [get\\_har\\_entry](#), [get\\_request\\_url](#), [get\\_response\\_body](#), [har\\_entry\\_count](#)

---

|                              |                             |
|------------------------------|-----------------------------|
| <code>get_request_url</code> | <i>Retrieve request URL</i> |
|------------------------------|-----------------------------|

---

**Description**

Retrieve request URL

**Usage**

```
get_request_url(har_resp_obj)
```

**Arguments**

`har_resp_obj` HAR response object

**See Also**

Other splash\_har\_helpers: [get\\_content\\_size](#), [get\\_content\\_type](#), [get\\_har\\_entry](#), [get\\_request\\_type](#), [get\\_response\\_body](#), [har\\_entry\\_count](#)



---

|                   |   |
|-------------------|---|
| get_response_body | <i>Retrieve the body content of a HAR entry</i> |
|-------------------|---|

---

**Description**

Retrieve the body content of a HAR entry

**Usage**

```
get_response_body(har_resp_obj, type = c("raw", "text"))
```

**Arguments**

|              |  |
|--------------|--|
| har_resp_obj | HAR response object  |
| type         | return type. If raw (default) then a raw vector of the content is returned. If text then a character vector. |

**Value**

A raw vector of the content or NULL or a character if type == text

**See Also**

Other splash\_har\_helpers: [get\\_content\\_size](#), [get\\_content\\_type](#), [get\\_har\\_entry](#), [get\\_request\\_type](#), [get\\_request\\_url](#), [har\\_entry\\_count](#)

---

|             |   |
|-------------|---|
| har_entries | <i>Retrieve just the HAR entries from a splashr request</i> |
|-------------|---|

---

**Description**

Retrieve just the HAR entries from a splashr request

**Usage**

```
har_entries(x)
```

**Arguments**

|   |  |
|---|--|
| x | can be a 'har' object, 'harlog' object or 'hareentries' object |
|---|--|

---

|                 |  |
|-----------------|--|
| har_entry_count | <i>Retrieves number of HAR entries in a response</i> |
|-----------------|--|

---

**Description**

Retrieves number of HAR entries in a response

**Usage**

```
har_entry_count(x)
```

**Arguments**

x can be a 'har' object, 'harlog' object or 'harentries' object

**See Also**

Other splash\_har\_helpers: [get\\_content\\_size](#), [get\\_content\\_type](#), [get\\_har\\_entry](#), [get\\_request\\_type](#), [get\\_request\\_url](#), [get\\_response\\_body](#)

---

|                |   |
|----------------|---|
| install_splash | <i>Retrieve the Docker image for Splash</i> |
|----------------|---|

---

**Description**

Retrieve the Docker image for Splash

**Usage**

```
install_splash(tag = "3.0")
```

**Arguments**

tag Splash Docker image tag to install

**See Also**

Other splash\_docker\_helpers: [start\\_splash](#), [stop\\_splash](#)

**Examples**

```
## Not run:  
install_splash()  
splash_container <- start_splash()  
stop_splash(splash_container)
```

```
## End(Not run)
```

---

|              |   |
|--------------|---|
| json_fromb64 | <i>Convert a Base64 encoded string into an R object</i> |
|--------------|---|

---

**Description**

A simple wrapper around calls to `openssl::base64_decode()` and `jsonlite::fromJSON()`.

**Usage**

```
json_fromb64(x, flatten = TRUE, ...)
```

**Arguments**

|         |  |
|---------|--|
| x       | a string                                       |
| flatten | flatten JSON structures upon conversion?       |
| ...     | passed on to <code>jsonlite::fromJSON()</code> |

---

|                |  |
|----------------|--|
| killall_splash | <i>Prune all dead and running Splash Docker containers</i> |
|----------------|--|

---

**Description**

This is a destructive function. It will stop **any** Docker container that is based on an image matching "scrapinghub/splashr". It's best used when you had a session forcefully interrupted and had been using the R helper functions to start/stop the Splash Docker container. You may want to consider using the Docker command-line interface to perform this work manually.

**Usage**

```
killall_splash()
```

---

|            |  |
|------------|--|
| render_har | <i>Return information about Splash interaction with a website in HAR format.</i> |
|------------|--|

---

**Description**

It includes information about requests made, responses received, timings, headers, etc and is incredibly detailed, full of information on every component loaded.

**Usage**

```
render_har(splash_obj = splash_local, url, base_url, response_body = FALSE,
  timeout = 30, resource_timeout, wait = 0, proxy, js, js_src, filters,
  allowed_domains, allowed_content_types, forbidden_content_types,
  viewport = "1024x768", images, headers, body, http_method, save_args,
  load_args)
```

**Arguments**

|                         |   |
|-------------------------|---|
| splash_obj              | Object created by a call to <a href="#">splash()</a>  |
| url                     | The URL to render (required)  |
| base_url                | The base url to render the page with.   |
| response_body           | When TRUE, response content is included in the HAR records  |
| timeout                 | A timeout (in seconds) for the render (defaults to 30). Without reconfiguring the startup parameters of the Splash server (not this package) the maximum allowed value for the timeout is 60 seconds. |
| resource_timeout        | A timeout (in seconds) for individual network requests.   |
| wait                    | Time (in seconds) to wait for updates after page is loaded (defaults to 0).   |
| proxy                   | Proxy profile name or proxy URL.  |
| js                      | Javascript profile name.  |
| js_src                  | JavaScript code to be executed in page context.   |
| filters                 | Comma-separated list of request filter names.   |
| allowed_domains         | Comma-separated list of allowed domain names. If present, Splash won't load anything neither from domains not in this list nor from subdomains of domains not in this list.                           |
| allowed_content_types   | Comma-separated list of allowed content types. If present, Splash will abort any request if the response's content type doesn't match any of the content types in this list. Wildcards are supported. |
| forbidden_content_types | Comma-separated list of forbidden content types. If present, Splash will abort any request if the response's content type matches any of the content types in this list. Wildcards are supported.     |
| viewport                | View width and height (in pixels) of the browser viewport to render the web page. Format is "<width>x<height>", e.g. 800x600. Default value is "full".  |
| images                  | Whether to download images.   |
| headers                 | HTTP headers to set for the first outgoing request.   |
| body                    | Body of HTTP POST request to be sent if method is POST.   |
| http_method             | HTTP method of outgoing Splash request.   |
| save_args               | A list of argument names to put in cache.   |
| load_args               | Parameter values to load from cache   |

**Value**

a [HARtools](#) har object

**References**

[Splash docs](#)

**See Also**

Other splash\_renderers: [execute\\_lua](#), [render\\_html](#), [render\\_jpeg](#), [render\\_json](#), [render\\_png](#)

---

|             |   |
|-------------|---|
| render_html | <i>Return the HTML of the javascript-rendered page.</i> |
|-------------|---|

---

**Description**

Similar (i.e. a dynamic equivalent) to `rvest::read_html`.

**Usage**

```
render_html(splash_obj = splash_local, url, base_url, timeout = 30,
  resource_timeout, wait = 0, proxy, js, js_src, filters, allowed_domains,
  allowed_content_types, forbidden_content_types, viewport = "1024x768",
  images, headers, body, http_method, save_args, load_args, raw_html = FALSE)
```

**Arguments**

|                  |   |
|------------------|---|
| splash_obj       | Object created by a call to <a href="#">splash()</a>  |
| url              | The URL to render (required)  |
| base_url         | The base url to render the page with.   |
| timeout          | A timeout (in seconds) for the render (defaults to 30). Without reconfiguring the startup parameters of the Splash server (not this package) the maximum allowed value for the timeout is 60 seconds. |
| resource_timeout | A timeout (in seconds) for individual network requests.   |
| wait             | Time (in seconds) to wait for updates after page is loaded (defaults to 0).   |
| proxy            | Proxy profile name or proxy URL.  |
| js               | Javascript profile name.  |
| js_src           | JavaScript code to be executed in page context.   |
| filters          | Comma-separated list of request filter names.   |
| allowed_domains  | Comma-separated list of allowed domain names. If present, Splash won't load anything neither from domains not in this list nor from subdomains of domains not in this list.                           |

|                         |   |
|-------------------------|---|
| allowed_content_types   | Comma-separated list of allowed content types. If present, Splash will abort any request if the response's content type doesn't match any of the content types in this list. Wildcards are supported. |
| forbidden_content_types | Comma-separated list of forbidden content types. If present, Splash will abort any request if the response's content type matches any of the content types in this list. Wildcards are supported.     |
| viewport                | View width and height (in pixels) of the browser viewport to render the web page. Format is "<width>x<height>", e.g. 800x600. Default value is "full".  |
| images                  | Whether to download images.   |
| headers                 | HTTP headers to set for the first outgoing request.   |
| body                    | Body of HTTP POST request to be sent if method is POST.   |
| http_method             | HTTP method of outgoing Splash request.   |
| save_args               | A list of argument names to put in cache.   |
| load_args               | Parameter values to load from cache   |
| raw_html                | if TRUE then return a character vector vs an XML document. Only valid for render_html   |

**Value**

An XML document. Note that this is processed by `xml2::read_html()` so it will not be the pristine, raw, rendered HTML from the site. Use `raw_html=TRUE` if you do not want it to be processed first by `xml2`. If you choose `raw_html=TRUE` you'll get back a character vector.

**References**

[Splash docs](#)

**See Also**

Other splash\_renderers: [execute\\_lua](#), [render\\_har](#), [render\\_jpeg](#), [render\\_json](#), [render\\_png](#)

---

render\_jpeg

*Return a image (in JPEG format) of the javascript-rendered page.*

---

**Description**

Return a image (in JPEG format) of the javascript-rendered page.

**Usage**

```
render_jpeg(splash_obj = splash_local, url, base_url = NULL, quality = 75,
            width, height, timeout = 30, resource_timeout, wait = 0,
            render_all = TRUE, proxy, js, js_src, filters, allowed_domains,
            allowed_content_types, forbidden_content_types, viewport = "full", images,
            headers, body, http_method, save_args, load_args)
```

**Arguments**

|                         |   |
|-------------------------|---|
| splash_obj              | Object created by a call to <a href="#">splash()</a>  |
| url                     | The URL to render (required)  |
| base_url                | The base url to render the page with.   |
| quality                 | JPEG quality parameter in range from 0 to 100. Default is quality=75.   |
| width                   | Resize the rendered image to the given width/height (in pixels) keeping the aspect ratio. These are optional  |
| height                  | Resize the rendered image to the given width/height (in pixels) keeping the aspect ratio. These are optional  |
| timeout                 | A timeout (in seconds) for the render (defaults to 30). Without reconfiguring the startup parameters of the Splash server (not this package) the maximum allowed value for the timeout is 60 seconds. |
| resource_timeout        | A timeout (in seconds) for individual network requests.   |
| wait                    | Time (in seconds) to wait for updates after page is loaded (defaults to 0).   |
| render_all              | If TRUE extend the viewport to include the whole webpage (possibly very tall) before rendering.   |
| proxy                   | Proxy profile name or proxy URL.  |
| js                      | Javascript profile name.  |
| js_src                  | JavaScript code to be executed in page context.   |
| filters                 | Comma-separated list of request filter names.   |
| allowed_domains         | Comma-separated list of allowed domain names. If present, Splash won't load anything neither from domains not in this list nor from subdomains of domains not in this list.                           |
| allowed_content_types   | Comma-separated list of allowed content types. If present, Splash will abort any request if the response's content type doesn't match any of the content types in this list. Wildcards are supported. |
| forbidden_content_types | Comma-separated list of forbidden content types. If present, Splash will abort any request if the response's content type matches any of the content types in this list. Wildcards are supported.     |
| viewport                | View width and height (in pixels) of the browser viewport to render the web page. Format is "<width>x<height>", e.g. 800x600. Default value is "full".  |
| images                  | Whether to download images.   |
| headers                 | HTTP headers to set for the first outgoing request.   |
| body                    | Body of HTTP POST request to be sent if method is POST.   |
| http_method             | HTTP method of outgoing Splash request.   |
| save_args               | A list of argument names to put in cache.   |
| load_args               | Parameter values to load from cache   |

**Value**

a [magick](#) image object

**References**

[Splash docs](#)

**See Also**

Other splash\_renderers: [execute\\_lua](#), [render\\_har](#), [render\\_html](#), [render\\_json](#), [render\\_png](#)

---

|             |   |
|-------------|---|
| render_json | <i>Return a json-encoded dictionary with information about javascript-rendered webpage.</i> |
|-------------|---|

---

**Description**

It can include HTML, PNG and other information, based on arguments passed.

**Usage**

```
render_json(splash_obj = splash_local, url, base_url = NULL, quality = 75,
            width, height, timeout = 30, resource_timeout, wait = 0,
            render_all = FALSE, proxy, js, js_src, filters, allowed_domains,
            allowed_content_types, forbidden_content_types, viewport = "1024x768",
            images, headers, body, http_method, save_args, load_args, html = TRUE,
            png = FALSE, jpeg = FALSE, iframes = TRUE, script = TRUE,
            console = TRUE, history = TRUE, har = TRUE, response_body = FALSE)
```

**Arguments**

|                  |   |
|------------------|---|
| splash_obj       | Object created by a call to <a href="#">splash()</a>  |
| url              | The URL to render (required)  |
| base_url         | The base url to render the page with.   |
| quality          | JPEG quality parameter in range from 0 to 100. Default is quality=75.   |
| width            | Resize the rendered image to the given width/height (in pixels) keeping the aspect ratio. These are optional  |
| height           | Resize the rendered image to the given width/height (in pixels) keeping the aspect ratio. These are optional  |
| timeout          | A timeout (in seconds) for the render (defaults to 30). Without reconfiguring the startup parameters of the Splash server (not this package) the maximum allowed value for the timeout is 60 seconds. |
| resource_timeout | A timeout (in seconds) for individual network requests.   |
| wait             | Time (in seconds) to wait for updates after page is loaded (defaults to 0).   |



|                         |  |
|-------------------------|--|
| render_all              | If TRUE extend the viewport to include the whole webpage (possibly very tall) before rendering.  |
| proxy                   | Proxy profile name or proxy URL.   |
| js                      | Javascript profile name.   |
| js_src                  | JavaScript code to be executed in page context.  |
| filters                 | Comma-separated list of request filter names.  |
| allowed_domains         | Comma-separated list of allowed domain names. If present, Splash won't load anything neither from domains not in this list nor from subdomains of domains not in this list.  |
| allowed_content_types   | Comma-separated list of allowed content types. If present, Splash will abort any request if the response's content type doesn't match any of the content types in this list. Wildcards are supported.  |
| forbidden_content_types | Comma-separated list of forbidden content types. If present, Splash will abort any request if the response's content type matches any of the content types in this list. Wildcards are supported.  |
| viewport                | View width and height (in pixels) of the browser viewport to render the web page. Format is "<width>x<height>", e.g. 800x600. Default value is "full".   |
| images                  | Whether to download images.  |
| headers                 | HTTP headers to set for the first outgoing request.  |
| body                    | Body of HTTP POST request to be sent if method is POST.  |
| http_method             | HTTP method of outgoing Splash request.  |
| save_args               | A list of argument names to put in cache.  |
| load_args               | Parameter values to load from cache  |
| html                    | Whether to include HTML in output.   |
| png                     | Whether to include PNG in output.  |
| jpeg                    | Whether to include JPEG in output.   |
| iframes                 | Whether to include information about child frames in output.   |
| script                  | Whether to include the result of the custom executed javascript final statement in output  |
| console                 | Whether to include the executed javascript console messages in output.   |
| history                 | Whether to include the history of requests/responses for webpage main frame. Use it to get HTTP status codes and headers. Only information about "main" requests/responses is returned (i.e. information about related resources like images and AJAX queries is not returned). To get information about all requests and responses use har parameter. |
| har                     | Whether to include HAR in output. If TRUE the result will contain the same data as <a href="#">render_har()</a> provides under har list entry. By default, response content is not included. To enable it use response_body parameter.   |
| response_body           | Used with har parameter.   |

**Value**

a huge list

**Note**

All "whether to include..." parameters are default TRUE except for png and jpeg and a custom print method is defined to stop your console from being overwhelmed with data. Use `str()` to inspect various portions of the result.

**References**

[Splash docs](#)

**See Also**

Other splash\_renderers: [execute\\_lua](#), [render\\_har](#), [render\\_html](#), [render\\_jpeg](#), [render\\_png](#)

---

|            |  |
|------------|--|
| render_png | <i>Return a image (in PNG format) of the javascript-rendered page.</i> |
|------------|--|

---

**Description**

Return a image (in PNG format) of the javascript-rendered page.

**Usage**

```
render_png(splash_obj = splash_local, url, base_url = NULL, width, height,
  timeout = 30, resource_timeout, wait = 0, render_all = TRUE, proxy, js,
  js_src, filters, allowed_domains, allowed_content_types,
  forbidden_content_types, viewport = "full", images, headers, body,
  http_method, save_args, load_args)
```

**Arguments**

|                  |   |
|------------------|---|
| splash_obj       | Object created by a call to <a href="#">splash()</a>  |
| url              | The URL to render (required)  |
| base_url         | The base url to render the page with.   |
| width, height    | Resize the rendered image to the given width/height (in pixels) keeping the aspect ratio. These are optional  |
| timeout          | A timeout (in seconds) for the render (defaults to 30). Without reconfiguring the startup parameters of the Splash server (not this package) the maximum allowed value for the timeout is 60 seconds. |
| resource_timeout | A timeout (in seconds) for individual network requests.   |
| wait             | Time (in seconds) to wait for updates after page is loaded (defaults to 0).   |

|                         |   |
|-------------------------|---|
| render_all              | If TRUE extend the viewport to include the whole webpage (possibly very tall) before rendering.   |
| proxy                   | Proxy profile name or proxy URL.  |
| js                      | Javascript profile name.  |
| js_src                  | JavaScript code to be executed in page context.   |
| filters                 | Comma-separated list of request filter names.   |
| allowed_domains         | Comma-separated list of allowed domain names. If present, Splash won't load anything neither from domains not in this list nor from subdomains of domains not in this list.                           |
| allowed_content_types   | Comma-separated list of allowed content types. If present, Splash will abort any request if the response's content type doesn't match any of the content types in this list. Wildcards are supported. |
| forbidden_content_types | Comma-separated list of forbidden content types. If present, Splash will abort any request if the response's content type matches any of the content types in this list. Wildcards are supported.     |
| viewport                | View width and height (in pixels) of the browser viewport to render the web page. Format is "<width>x<height>", e.g. 800x600. Default value is "full".  |
| images                  | Whether to download images.   |
| headers                 | HTTP headers to set for the first outgoing request.   |
| body                    | Body of HTTP POST request to be sent if method is POST.   |
| http_method             | HTTP method of outgoing Splash request.   |
| save_args               | A list of argument names to put in cache.   |
| load_args               | Parameter values to load from cache   |

**Value**

a [magick](#) image object

**References**

[Splash docs](#)

**See Also**

Other splash\_renderers: [execute\\_lua](#), [render\\_har](#), [render\\_html](#), [render\\_jpeg](#), [render\\_json](#)

**Examples**

```
## Not run:
render_png(url = "https://httpbin.org/")

## End(Not run)
```

---

|        |   |
|--------|---|
| splash | <i>Configure parameters for connecting to a Splash server</i> |
|--------|---|

---

**Description**

Configure parameters for connecting to a Splash server

**Usage**

```
splash(host, port = 8050L)
```

```
splash_local
```

**Arguments**

|      |   |
|------|---|
| host | host or IP address                              |
| port | port the server is running on (default is 8050) |

**Format**

An object of class `list` of length 2.

**Examples**

```
## Not run:
sp <- splash()

## End(Not run)
```

---

|         |   |
|---------|---|
| splashr | <i>Tools to Work with the 'Splash' JavaScript Rendering Service</i> |
|---------|---|

---

**Description**

'Splash' <https://github.com/scrapinghub/splash> is a 'JavaScript' rendering service. It's a lightweight web browser with an 'HTTP' API, implemented in 'Python' using 'Twisted' and 'QT' and provides some of the core functionality of the 'RSelenium' or 'seleniumPipes' R packages in a lightweight footprint.

**Details**

Some of 'Splash' features include the ability to process multiple webpages in parallel; retrieving 'HTML' results and/or take screenshots; disabling images or use 'Adblock Plus' rules to make rendering faster; executing custom 'JavaScript' in page context; getting detailed rendering info in 'HAR' format.

**Author(s)**

Bob Rudis (bob@rud.is)

---

|                 |                                   |
|-----------------|-----------------------------------|
| splashr-exports | <i>splashr exported operators</i> |
|-----------------|-----------------------------------|

---

**Description**

The following functions are imported and then re-exported from the splashr package to enable use of the magrittr pipe operator with no additional library calls

---

|               |                                      |
|---------------|--------------------------------------|
| splash_active | <i>Test if a Splash server is up</i> |
|---------------|--------------------------------------|

---

**Description**

Test if a Splash server is up

**Usage**

```
splash_active(splash_obj = splash_local)
```

**Arguments**

splash\_obj      A splash connection object

**Value**

TRUE if Splash server is running, otherwise FALSE

**See Also**

Other splash\_info\_functions: [splash\\_debug](#), [splash\\_history](#), [splash\\_perf\\_stats](#), [splash\\_version](#)

**Examples**

```
## Not run:  
sp <- splash()  
splash_active(sp)  
  
## End(Not run)
```

---

|                |   |
|----------------|---|
| splash_add_lua | <i>Add raw lua code into DSL call chain</i> |
|----------------|---|

---

### Description

The splashr lua DSL (domain specific language) wrapper wraps what the package author believes to be the most common/useful lua functions. Users of the package may have need to insert some custom lua code within a DSL call chain they are building. You can insert any Splash lua code you like with this function call.

### Usage

```
splash_add_lua(splash_obj, lua_code)
```

### Arguments

|            |   |
|------------|---|
| splash_obj | splashr object                            |
| lua_code   | length 1 character vector of raw lua code |

### Details

The code is inserted at the position the `splash_add_lua()` is called in the chain which will be within the main "splash" function which is defined as:

```
function main(splash)
  ...
end
```

If you need more flexibility, use the `execute_lua()` function.

### See Also

Other splash\_dsl\_functions: [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

|              |   |
|--------------|---|
| splash_click | <i>Trigger mouse click event in web page.</i> |
|--------------|---|

---

### Description

Trigger mouse click event in web page.

### Usage

```
splash_click(splash_obj, x, y)
```

**Arguments**

|            |  |
|------------|--|
| splash_obj | splashr object   |
| x, y       | coordinates (distances from the left or top, relative to the current viewport) |

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

splash\_enable\_javascript

*Enable or disable execution of JavaScript code embedded in the page.*

---

**Description**

JavaScript execution is enabled by default.

**Usage**

```
splash_enable_javascript(splash_obj, enable = TRUE)
```

**Arguments**

|            |                |
|------------|----------------|
| splash_obj | splashr object |
| enable     | logical        |

**See Also**

Other splash\_dsl\_attributes: [splash\\_images](#), [splash\\_plugins](#), [splash\\_private\\_mode](#), [splash\\_response\\_body](#)

**Examples**

```
## Not run:
splash_local %>%
  splash_response_body(TRUE) %>%
  splash_private_mode(TRUE) %>%
  splash_enable_javascript(FALSE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har

## End(Not run)
```

---

|              |   |
|--------------|---|
| splash_focus | <i>Focus on a document element provided by a CSS selector</i> |
|--------------|---|

---

**Description**

Focus on a document element provided by a CSS selector

**Usage**

```
splash_focus(splash_obj, selector)
```

**Arguments**

|            |                    |
|------------|--------------------|
| splash_obj | splashr object     |
| selector   | valid CSS selector |

**References**

See [the docs](#) for more info

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

|           |                      |
|-----------|----------------------|
| splash_go | <i>Go to an URL.</i> |
|-----------|----------------------|

---

**Description**

This is similar to entering an URL in a browser address bar, pressing Enter and waiting until page loads.

**Usage**

```
splash_go(splash_obj, url)
```

**Arguments**

|            |                |
|------------|----------------|
| splash_obj | splashr object |
| url        | - URL to load; |



**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

**Examples**

```
## Not run:
splash_local %>%
  splash_response_body(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har

## End(Not run)
```

---

|            |  |
|------------|--|
| splash_har | <i>Return information about Splash interaction with a website in HAR format.</i> |
|------------|--|

---

**Description**

Similar to [render\\_har\(\)](#) but used in a script context. Should be the LAST element in a DSL script chain as this will execute the script and return the HAR content

**Usage**

```
splash_har(splash_obj)
```

**Arguments**

```
splash_obj    splashr object
```

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

**Examples**

```
## Not run:
splash_local %>%
  splash_response_body(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har
```

```
## End(Not run)
```

---

|                  |   |
|------------------|---|
| splash_har_reset | <i>Drops all internally stored HAR records.</i> |
|------------------|---|

---

**Description**

Drops all internally stored HAR records.

**Usage**

```
splash_har_reset(splash_obj)
```

**Arguments**

splash\_obj      splashr object

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

|                |  |
|----------------|--|
| splash_history | <i>Get information about requests/responses for the pages loaded</i> |
|----------------|--|

---

**Description**

Get information about requests/responses for the pages loaded

**Usage**

```
splash_history(splash_obj = splash_local)
```

**Arguments**

splash\_obj      A splash connection object

**See Also**

Other splash\_info\_functions: [splash\\_active](#), [splash\\_debug](#), [splash\\_perf\\_stats](#), [splash\\_version](#)

## Examples

```
## Not run:  
sp <- splash()  
splash_history(sp)  
  
## End(Not run)
```

---

|             |  |
|-------------|--|
| splash_html | <i>Return a HTML snapshot of a current page.</i> |
|-------------|--|

---

## Description

Similar to [render\\_html\(\)](#) but used in a script context. Should be the LAST element in a DSL script chain as this will execute the script and return the HTML content

## Usage

```
splash_html(splash_obj, raw_html = FALSE)
```

## Arguments

|            |  |
|------------|--|
| splash_obj | splashr object   |
| raw_html   | if TRUE then return a character vector vs an XML document. |

## See Also

Other `splash_dsl_functions`: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

## Examples

```
## Not run:  
splash_local %>%  
  splash_response_body(TRUE) %>%  
  splash_user_agent(ua_macos_chrome) %>%  
  splash_go("https://rud.is/b") %>%  
  splash_wait(2) %>%  
  splash_html() -> rud_pg  
  
## End(Not run)
```

---

|               |                              |
|---------------|------------------------------|
| splash_images | <i>Enable/disable images</i> |
|---------------|------------------------------|

---

### Description

By default, images are enabled. Disabling of the images can save a lot of network traffic (usually around ~50 affect the JavaScript code inside page: disabling of the images may change sizes and positions of DOM elements, and scripts may read and use them.

### Usage

```
splash_images(splash_obj, enable = TRUE)
```

### Arguments

|            |                |
|------------|----------------|
| splash_obj | splashr object |
| enable     | logical        |

### See Also

Other splash\_dsl\_attributes: [splash\\_enable\\_javascript](#), [splash\\_plugins](#), [splash\\_private\\_mode](#), [splash\\_response\\_body](#)

### Examples

```
## Not run:
splash_local %>%
  splash_images(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har

## End(Not run)
```

---

|                   |  |
|-------------------|--|
| splash_perf_stats | <i>Get Splash performance-related statistics</i> |
|-------------------|--|

---

### Description

Get Splash performance-related statistics

### Usage

```
splash_perf_stats(splash_obj = splash_local)
```

**Arguments**

splash\_obj      A splash connection object

**See Also**

Other splash\_info\_functions: [splash\\_active](#), [splash\\_debug](#), [splash\\_history](#), [splash\\_version](#)

**Examples**

```
## Not run:  
sp <- splash()  
splash_perf_stats(sp)  
  
## End(Not run)
```

---

|                |  |
|----------------|--|
| splash_plugins | <i>Enable or disable browser plugins (e.g. Flash).</i> |
|----------------|--|

---

**Description**

Plugins are disabled by default.

**Usage**

```
splash_plugins(splash_obj, enable = FALSE)
```

**Arguments**

splash\_obj      splashr object  
enable           logical

**See Also**

Other splash\_dsl\_attributes: [splash\\_enable\\_javascript](#), [splash\\_images](#), [splash\\_private\\_mode](#), [splash\\_response\\_body](#)

**Examples**

```
## Not run:  
splash_local %>%  
  splash_plugins(TRUE) %>%  
  splash_user_agent(ua_macos_chrome) %>%  
  splash_go("https://rud.is/b") %>%  
  splash_wait(2) %>%  
  splash_har() -> rud_har  
  
## End(Not run)
```

---

|            |   |
|------------|---|
| splash_png | <i>Return a screenshot of a current page in PNG format.</i> |
|------------|---|

---

**Description**

Similar to [render\\_png\(\)](#) but used in a script context. Should be the LAST element in a DSL script chain as this will execute the script and return the PNG content

**Usage**

```
splash_png(splash_obj)
```

**Arguments**

```
splash_obj    splashr object
```

**Value**

a [magick](#) image object

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

**Examples**

```
## Not run:
splash_local %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_png()

## End(Not run)
```

---

|              |   |
|--------------|---|
| splash_press | <i>Trigger mouse press event in web page.</i> |
|--------------|---|

---

**Description**

Trigger mouse press event in web page.

**Usage**

```
splash_press(splash_obj, x, y)
```

**Arguments**

|            |  |
|------------|--|
| splash_obj | splashr object   |
| x, y       | coordinates (distances from the left or top, relative to the current viewport) |

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

|                     |   |
|---------------------|---|
| splash_private_mode | <i>Enable or disable execution of JavaScript code embedded in the page.</i> |
|---------------------|---|

---

**Description**

Private mode is enabled by default unless you pass flag `--disable-private-mode` at Splash (server) startup. Note that if you disable private mode browsing data such as cookies or items kept in local storage may persist between requests.

**Usage**

```
splash_private_mode(splash_obj, enable = FALSE)
```

**Arguments**

|            |                |
|------------|----------------|
| splash_obj | splashr object |
| enable     | logical        |

**See Also**

Other splash\_dsl\_attributes: [splash\\_enable\\_javascript](#), [splash\\_images](#), [splash\\_plugins](#), [splash\\_response\\_body](#)

**Examples**

```
## Not run:
splash_local %>%
  splash_response_body(TRUE) %>%
  splash_private_mode(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har

## End(Not run)
```

---

splash\_release      *Trigger mouse release event in web page.*

---

### Description

Trigger mouse release event in web page.

### Usage

```
splash_release(splash_obj, x, y)
```

### Arguments

|            |  |
|------------|--|
| splash_obj | splashr object   |
| x, y       | coordinates (distances from the left or top, relative to the current viewport) |

### See Also

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

splash\_response\_body      *Enable or disable response content tracking.*

---

### Description

By default Splash doesn't keep bodies of each response in memory, for efficiency reasons.

### Usage

```
splash_response_body(splash_obj, enable = FALSE)
```

### Arguments

|            |                |
|------------|----------------|
| splash_obj | splashr object |
| enable     | logical        |

### See Also

Other splash\_dsl\_attributes: [splash\\_enable\\_javascript](#), [splash\\_images](#), [splash\\_plugins](#), [splash\\_private\\_mode](#)



## Examples

```
## Not run:
splash_local %>%
  splash_response_body(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har

## End(Not run)
```

---

|                  |  |
|------------------|--|
| splash_send_keys | <i>Send keyboard events to page context.</i> |
|------------------|--|

---

## Description

- whitespace is ignored and only used to separate the different keys
- characters are literally represented

## Usage

```
splash_send_keys(splash_obj, keys)
```

## Arguments

|            |                |
|------------|----------------|
| splash_obj | splshr object  |
| keys       | string to send |

## Details

This is different from [splash\\_send\\_text\(\)](#)

## References

See [the docs](#) for more info

## See Also

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_text](#), [splash\\_wait](#)

---

|                  |   |
|------------------|---|
| splash_send_text | <i>Send text as input to page context, literally, character by character.</i> |
|------------------|---|

---

### Description

This is different from [splash\\_send\\_keys\(\)](#)

### Usage

```
splash_send_text(splash_obj, text)
```

### Arguments

|            |                |
|------------|----------------|
| splash_obj | splashr object |
| text       | string to send |

### Note

This adds a call to `splash:wait` so you do not have to

### References

See [the docs](#) for more info

### See Also

Other `splash_dsl` functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_wait](#)

---

|                   |  |
|-------------------|--|
| splash_user_agent | <i>Overwrite the User-Agent header for all further requests.</i> |
|-------------------|--|

---

### Description

There are a few built-in user agents, all beginning with `ua_`.

**Usage**

```
splash_user_agent(splash_obj, user_agent = ua_splashr)
```

```
ua_splashr
```

```
ua_win10_chrome
```

```
ua_win10_firefox
```

```
ua_win10_ie11
```

```
ua_win7_chrome
```

```
ua_win7_firefox
```

```
ua_win7_ie11
```

```
ua_macos_chrome
```

```
ua_macos_safari
```

```
ua_linux_chrome
```

```
ua_linux_firefox
```

```
ua_ios_safari
```

**Arguments**

```
splash_obj    splashr object
```

```
user_agent    1 element character vector, defaults to splashr/#.#.#.
```

**Format**

An object of class character of length 1.

**Examples**

```
## Not run:  
library(rvest)  
  
URL <- "https://httpbin.org/user-agent"  
  
splash_local %>%  
  splash_response_body(TRUE) %>%  
  splash_user_agent(ua_macos_chrome) %>%  
  splash_go(URL) %>%  
  splash_html() %>%  
  html_text("body") %>%
```

```

    jsonlite::fromJSON()
## End(Not run)

```

---

|                |                                       |
|----------------|---------------------------------------|
| splash_version | <i>Get Splash version information</i> |
|----------------|---------------------------------------|

---

### Description

Get Splash version information

### Usage

```
splash_version(splash_obj = splash_local)
```

### Arguments

splash\_obj      A splash connection object

### See Also

Other splash\_info\_functions: [splash\\_active](#), [splash\\_debug](#), [splash\\_history](#), [splash\\_perf\\_stats](#)

### Examples

```

## Not run:
sp <- splash()
splash_version(sp)

## End(Not run)

```

---

|             |                               |
|-------------|-------------------------------|
| splash_wait | <i>Wait for a period time</i> |
|-------------|-------------------------------|

---

### Description

When script is waiting WebKit continues processing the webpage

### Usage

```
splash_wait(splash_obj, time = 2)
```

### Arguments

splash\_obj      splashr object  
time              number of seconds to wait

**See Also**

Other splash\_dsl\_functions: [splash\\_add\\_lua](#), [splash\\_click](#), [splash\\_focus](#), [splash\\_go](#), [splash\\_har\\_reset](#), [splash\\_har](#), [splash\\_html](#), [splash\\_png](#), [splash\\_press](#), [splash\\_release](#), [splash\\_send\\_keys](#), [splash\\_send\\_text](#)

**Examples**

```
## Not run:
splash_local %>%
  splash_response_body(TRUE) %>%
  splash_user_agent(ua_macos_chrome) %>%
  splash_go("https://rud.is/b") %>%
  splash_wait(2) %>%
  splash_har() -> rud_har

## End(Not run)
```

---

|              |   |
|--------------|---|
| start_splash | <i>Start a Splash server Docker container</i> |
|--------------|---|

---

**Description**

If using this in an automation context, you should consider adding a `'Sys.sleep(3)'` (or higher) after starting the docker container.

**Usage**

```
start_splash(tag = "3.0")
```

**Arguments**

tag                      Splash Docker image tag to start

**Value**

'docker' 'container' object

**Note**

you need Docker running on your system and have pulled the container with `[install_splash]` for this to work. You should save the resultant object for use in `[stop_splash]` otherwise you'll have to kill it from the command line interface.

**See Also**

Other splash\_docker\_helpers: [install\\_splash](#), [stop\\_splash](#)

**Examples**

```
## Not run:
install_splash()
splash_container <- start_splash()
stop_splash(splash_container)

## End(Not run)
```

---

`stop_splash`*Stop a running a Splash server Docker container*

---

**Description**

Stop a running a Splash server Docker container

**Usage**

```
stop_splash(splash_container)
```

**Arguments**

`splash_container`  
Docker ‘container’ object created by [`start_splash()`]

**Note**

you need Docker running on your system and have pulled the container with [`install_splash()`] and started the Splash container with [`start_splash()`] for this to work. You will need the ‘container’ object from [`start_splash()`] for this to work.

**See Also**

Other splash\_docker\_helpers: [install\\_splash](#), [start\\_splash](#)

**Examples**

```
## Not run:
install_splash()
splash_container <- start_splash()
stop_splash(splash_container)

## End(Not run)
```

# Index

## \*Topic **datasets**

- splash, 20
- splash\_user\_agent, 34
- %>(splashtools-exports), 21
  
- as\_har, 3
- as\_http\_req, 3
- as\_response, 4
  
- execute\_lua, 4, 13, 14, 16, 18, 19
- execute\_lua(), 22
  
- get\_body\_size (get\_content\_size), 6
- get\_content\_size, 6, 7–10
- get\_content\_type, 6, 6, 7–10
- get\_har\_entry, 6, 7, 7, 8–10
- get\_headers\_size (get\_content\_size), 6
- get\_request\_type, 6–8, 8, 9, 10
- get\_request\_url, 6–8, 8, 9, 10
- get\_response\_body, 6–8, 9, 10
  
- har\_entries, 9
- har\_entry\_count, 6–9, 10
- HARtools, 13
- HARviewer (splashtools-exports), 21
- HARviewerOutput (splashtools-exports), 21
  
- install\_splash, 10, 37, 38
- is\_binary (get\_content\_type), 6
- is\_content\_type (get\_content\_type), 6
- is\_css (get\_content\_type), 6
- is\_get (get\_request\_type), 8
- is\_gif (get\_content\_type), 6
- is\_html (get\_content\_type), 6
- is\_javascript (get\_content\_type), 6
- is\_jpeg (get\_content\_type), 6
- is\_json (get\_content\_type), 6
- is\_plain (get\_content\_type), 6
- is\_png (get\_content\_type), 6
- is\_post (get\_request\_type), 8
- is\_svg (get\_content\_type), 6
  
- is\_xhr (get\_content\_type), 6
- is\_xml (get\_content\_type), 6
  
- json\_fromb64, 11
  
- killall\_splash, 11
  
- magick, 16, 19, 30
  
- render\_har, 5, 11, 14, 16, 18, 19
- render\_har(), 17, 25
- render\_html, 5, 13, 13, 16, 18, 19
- render\_html(), 27
- render\_jpeg, 5, 13, 14, 14, 18, 19
- render\_json, 5, 13, 14, 16, 16, 19
- render\_png, 5, 13, 14, 16, 18, 18
- render\_png(), 30
- renderHARviewer (splashtools-exports), 21
  
- splash, 20
- splash(), 5, 12, 13, 15, 16, 18
- splash\_active, 21, 26, 29, 36
- splash\_add\_lua, 22, 23–27, 30–34, 37
- splash\_click, 22, 22, 24–27, 30–34, 37
- splash\_debug, 21, 26, 29, 36
- splash\_enable\_javascript, 23, 28, 29, 31, 32
- splash\_focus, 22, 23, 24, 25–27, 30–34, 37
- splash\_go, 22–24, 24, 25–27, 30–34, 37
- splash\_har, 22–25, 25, 26, 27, 30–34, 37
- splash\_har\_reset, 22–25, 26, 27, 30–34, 37
- splash\_history, 21, 26, 29, 36
- splash\_html, 22–26, 27, 30–34, 37
- splash\_images, 23, 28, 29, 31, 32
- splash\_local (splash), 20
- splash\_perf\_stats, 21, 26, 28, 36
- splash\_plugins, 23, 28, 29, 31, 32
- splash\_png, 22–27, 30, 31–34, 37
- splash\_press, 22–27, 30, 30, 32–34, 37
- splash\_private\_mode, 23, 28, 29, 31, 32
- splash\_release, 22–27, 30, 31, 32, 33, 34, 37

`splash_response_body`, [23](#), [28](#), [29](#), [31](#), [32](#)  
`splash_send_keys`, [22–27](#), [30–32](#), [33](#), [34](#), [37](#)  
`splash_send_keys()`, [34](#)  
`splash_send_text`, [22–27](#), [30–33](#), [34](#), [37](#)  
`splash_send_text()`, [33](#)  
`splash_user_agent`, [34](#)  
`splash_version`, [21](#), [26](#), [29](#), [36](#)  
`splash_wait`, [22–27](#), [30–34](#), [36](#)  
`splashr`, [20](#)  
`splashr-exports`, [21](#)  
`splashr-package (splashr)`, [20](#)  
`start_splash`, [10](#), [37](#), [38](#)  
`stop_splash`, [10](#), [37](#), [38](#)  
`str()`, [18](#)

`ua_ios_safari (splash_user_agent)`, [34](#)  
`ua_linux_chrome (splash_user_agent)`, [34](#)  
`ua_linux_firefox (splash_user_agent)`, [34](#)  
`ua_macos_chrome (splash_user_agent)`, [34](#)  
`ua_macos_safari (splash_user_agent)`, [34](#)  
`ua_splashr (splash_user_agent)`, [34](#)  
`ua_win10_chrome (splash_user_agent)`, [34](#)  
`ua_win10_firefox (splash_user_agent)`, [34](#)  
`ua_win10_ie11 (splash_user_agent)`, [34](#)  
`ua_win7_chrome (splash_user_agent)`, [34](#)  
`ua_win7_firefox (splash_user_agent)`, [34](#)  
`ua_win7_ie11 (splash_user_agent)`, [34](#)

`writeHAR (splashr-exports)`, [21](#)

`xml2::read_html()`, [14](#)