Package 'webshot'

July 22, 2025

Title Take Screenshots of Web Pages

Version 0.5.5

Description Takes screenshots of web pages, including Shiny applications and R Markdown documents.

Depends R (>= 3.0)

Imports magrittr, jsonlite, callr

Suggests httpuv, knitr, rmarkdown, shiny, testthat (>= 3.0.0)

License GPL-2

SystemRequirements PhantomJS for taking screenshots, ImageMagick or GraphicsMagick and OptiPNG for manipulating images.

RoxygenNote 7.2.3

Encoding UTF-8

URL https://wch.github.io/webshot/, https://github.com/wch/webshot/

BugReports https://github.com/wch/webshot/issues

Config/testthat/edition 3

NeedsCompilation no

Author Winston Chang [aut, cre],
Yihui Xie [ctb],
Francois Guillem [ctb],
Barret Schloerke [ctb],
Nicolas Perriault [ctb] (The CasperJS library)

Maintainer Winston Chang <winston@rstudio.com>

Repository CRAN

Date/Publication 2023-06-26 23:30:02 UTC

Contents

appshot 2
install_phantomjs 3

is_phantomjs_installed 4
resize 5
rmdshot 5
shrink 7
webshot 7

Index **11**

`appshot` *Take a screenshot of a Shiny app*

Description

`appshot` performs a `webshot` using two different methods depending upon the object provided. If a 'character' is provided (pointing to an app.R file or app directory) an isolated background R process is launched to run the Shiny application. The current R process then captures the `webshot`. When a Shiny application object is supplied to `appshot`, it is reversed: the Shiny application runs in the current R process and an isolated background R process is launched to capture a `webshot`. The reason it is reversed in the second case has to do with scoping: although it would be preferable to run the Shiny application in a background process and call `webshot` from the current process, with Shiny application objects, there are potential scoping errors when run this way.

Usage

```
appshot(
  app,
  file = "webshot.png",
  ...,
  port = getOption("shiny.port"),
  envvars = NULL
)

## S3 method for class 'character'
appshot(
  app,
  file = "webshot.png",
  ...,
  port = getOption("shiny.port"),
  envvars = NULL
)

## S3 method for class 'shiny.appobj'
appshot(
  app,
  file = "webshot.png",
  ...,
  port = getOption("shiny.port"),
  envvars = NULL,
```

```
  webshot_timeout = 60
)
```

Arguments

app	A Shiny app object, or a string naming an app directory.
file	A vector of names of output files. Should end with .png, .pdf, or .jpeg. If several screenshots have to be taken and only one filename is provided, then the function appends the index number of the screenshot to the file name.
...	Other arguments to pass on to webshot.
port	Port that Shiny will listen on.
envvars	A named character vector or named list of environment variables and values to set for the Shiny app's R process. These will be unset after the process exits. This can be used to pass configuration information to a Shiny app.
webshot_timeout	The maximum number of seconds the phantom application is allowed to run before killing the process. If a delay argument is supplied (in ...), the delay value is added to the timeout value.

Examples

```
if (interactive()) {
  appdir <- system.file("examples", "01_hello", package="shiny")

  # With a Shiny directory
  appshot(appdir, "01_hello.png")

  # With a Shiny App object
  shinyapp <- shiny::shinyAppDir(appdir)
  appshot(shinyapp, "01_hello_app.png")
}
```

---

install_phantomjs *Install PhantomJS*

Description

Download the zip package, unzip it, and copy the executable to a system directory in which **webshot** can look for the PhantomJS executable.

Usage

```
install_phantomjs(
  version = "2.1.1",
  baseURL = "https://github.com/wch/webshot/releases/download/v0.3.1/",
  force = FALSE
)
```

Arguments

version	The version number of PhantomJS.
baseURL	The base URL for the location of PhantomJS binaries for download. If the default download site is unavailable, you may specify an alternative mirror, such as "https://bitbucket.org/ariya/phantomjs/downloads/".
force	Install PhantomJS even if the version installed is the latest or if the requested version is older. This is useful to reinstall or downgrade the version of PhantomJS.

Details

This function was designed primarily to help Windows users since it is cumbersome to modify the PATH variable. Mac OS X users may install PhantomJS via Homebrew. If you download the package from the PhantomJS website instead, please make sure the executable can be found via the PATH variable.

On Windows, the directory specified by the environment variable APPDATA is used to store 'phantomjs.exe'. On OS X, the directory '~/Library/Application Support' is used. On other platforms (such as Linux), the directory '~/bin' is used. If these directories are not writable, the directory 'PhantomJS' under the installation directory of the **webshot** package will be tried. If this directory still fails, you will have to install PhantomJS by yourself.

If PhantomJS is not already installed on the computer, this function will attempt to install it. However, if the version of PhantomJS installed is greater than or equal to the requested version, this function will not perform the installation procedure again unless the force parameter is set to TRUE. As a result, this function may also be used to reinstall or downgrade the version of PhantomJS found.

Value

NULL (the executable is written to a system directory).

---

is_phantomjs_installed

Determine if PhantomJS is Installed

Description

Verifies that a version of PhantomJS is installed and available for use on the user's computer.

Usage

```
is_phantomjs_installed()
```

Value

TRUE if the PhantomJS is installed. Otherwise, FALSE if PhantomJS is not installed.

resize *Resize an image*

Description

This does not change size of the image in pixels, nor does it affect appearance – it is lossless compression. This requires GraphicsMagick (recommended) or ImageMagick to be installed.

Usage

```
resize(filename, geometry)
```

Arguments

filename	Character vector containing the path of images to resize.
geometry	Scaling specification. Can be a percent, as in "50%", or pixel dimensions like "120x120", "120x", or "x120". Any valid ImageMagick geometry specifation can be used. If filename contains multiple images, this can be a vector to specify distinct sizes for each image.

Examples

```
if (interactive()) {
  # Can be chained with webshot() or appshot()
  webshot("https://www.r-project.org/", "r-small-1.png") %>%
    resize("75%")

  # Generate image that is 400 pixels wide
  webshot("https://www.r-project.org/", "r-small-2.png") %>%
    resize("400x")
}
```

rmdshot *Take a snapshot of an R Markdown document*

Description

This function can handle both static Rmd documents and Rmd documents with runtime: shiny.

Usage

```
rmdshot(
  doc,
  file = "webshot.png",
  ...,
  delay = NULL,
  rmd_args = list(),
  port = getOption("shiny.port"),
  envvars = NULL
)
```

Arguments

doc	The path to a Rmd document.
file	A vector of names of output files. Should end with .png, .pdf, or .jpeg. If several screenshots have to be taken and only one filename is provided, then the function appends the index number of the screenshot to the file name.
...	Other arguments to pass on to webshot.
delay	Time to wait before taking screenshot, in seconds. Sometimes a longer delay is needed for all assets to display properly. If NULL (the default), then it will use 0.2 seconds for static Rmd documents, and 3 seconds for Rmd documents with runtime:shiny.
rmd_args	A list of additional arguments to pass to either render (for static Rmd documents) or run (for Rmd documents with runtime:shiny).
port	Port that Shiny will listen on.
envvars	A named character vector or named list of environment variables and values to set for the Shiny app's R process. These will be unset after the process exits. This can be used to pass configuration information to a Shiny app.

Examples

```
if (interactive()) {
  # rmdshot("rmarkdown_file.Rmd", "snapshot.png")

  # R Markdown file
  input_file <- system.file("examples/knitr-minimal.Rmd", package = "knitr")
  rmdshot(input_file, "minimal_rmd.png")

  # Shiny R Markdown file
  input_file <- system.file("examples/shiny.Rmd", package = "webshot")
  rmdshot(input_file, "shiny_rmd.png", delay = 5)
}
```

shrink *Shrink file size of a PNG*

Description

This does not change size of the image in pixels, nor does it affect appearance – it is lossless compression. This requires the program `optipng` to be installed.

Usage

```
shrink(filename)
```

Arguments

`filename`	Character vector containing the path of images to resize. Must be PNG files.

Details

If other operations like resizing are performed, shrinking should occur as the last step. Otherwise, if the resizing happens after file shrinking, it will be as if the shrinking didn't happen at all.

Examples

```
if (interactive()) {
  webshot("https://www.r-project.org/", "r-shrink.png") %>%
    shrink()
}
```

webshot *Take a screenshot of a URL*

Description

Take a screenshot of a URL

Usage

```
webshot(
  url = NULL,
  file = "webshot.png",
  vwidth = 992,
  vheight = 744,
  cliprect = NULL,
  selector = NULL,
  expand = NULL,
  delay = 0.2,
```

```
  zoom = 1,
  eval = NULL,
  debug = FALSE,
  useragent = NULL
)
```

Arguments

`url`	A vector of URLs to visit.
`file`	A vector of names of output files. Should end with `.png`, `.pdf`, or `.jpeg`. If several screenshots have to be taken and only one filename is provided, then the function appends the index number of the screenshot to the file name.
`vwidth`	Viewport width. This is the width of the browser "window".
`vheight`	Viewport height This is the height of the browser "window".
`cliprect`	Clipping rectangle. If `cliprect` and `selector` are both unspecified, the clipping rectangle will contain the entire page. This can be the string `"viewport"`, in which case the clipping rectangle matches the viewport size, or it can be a four-element numeric vector specifying the top, left, width, and height. When taking screenshots of multiple URLs, this parameter can also be a list with same length as `url` with each element of the list being "viewport" or a four-elements numeric vector. This option is not compatible with `selector`.
`selector`	One or more CSS selectors specifying a DOM element to set the clipping rectangle to. The screenshot will contain these DOM elements. For a given selector, if it has more than one match, only the first one will be used. This option is not compatible with `cliprect`. When taking screenshots of multiple URLs, this parameter can also be a list with same length as `url` with each element of the list containing a vector of CSS selectors to use for the corresponding URL.
`expand`	A numeric vector specifying how many pixels to expand the clipping rectangle by. If one number, the rectangle will be expanded by that many pixels on all sides. If four numbers, they specify the top, right, bottom, and left, in that order. When taking screenshots of multiple URLs, this parameter can also be a list with same length as `url` with each element of the list containing a single number or four numbers to use for the corresponding URL.
`delay`	Time to wait before taking screenshot, in seconds. Sometimes a longer delay is needed for all assets to display properly.
`zoom`	A number specifying the zoom factor. A zoom factor of 2 will result in twice as many pixels vertically and horizontally. Note that using 2 is not exactly the same as taking a screenshot on a HiDPI (Retina) device: it is like increasing the zoom to 200 doubling the height and width of the browser window. This differs from using a HiDPI device because some web pages load different, higher-resolution images when they know they will be displayed on a HiDPI device (but using zoom will not report that there is a HiDPI device).
`eval`	An optional string with JavaScript code which will be evaluated after opening the page and waiting for `delay`, but before calculating the clipping region and taking the screenshot. See the Casper API for more information about what commands can be used to control the web page. NOTE: This is experimental and likely to change!

debug	Print out debugging messages from PhantomJS and CasperJS. This can help to diagnose problems.
useragent	The User-Agent header used to request the URL. Changing the User-Agent can mitigate rendering issues for some websites.

See Also

appshot for taking screenshots of Shiny applications.

Examples

```
if (interactive()) {

# Whole web page
webshot("https://github.com/rstudio/shiny")

# Might need a longer delay for all assets to display
webshot("http://rstudio.github.io/leaflet", delay = 0.5)

# One can also take screenshots of several URLs with only one command.
# This is more efficient than calling 'webshot' multiple times.
webshot(c("https://github.com/rstudio/shiny",
          "http://rstudio.github.io/leaflet"),
        delay = 0.5)

# Clip to the viewport
webshot("http://rstudio.github.io/leaflet", "leaflet-viewport.png",
        cliprect = "viewport")

# Manual clipping rectangle
webshot("http://rstudio.github.io/leaflet", "leaflet-clip.png",
        cliprect = c(200, 5, 400, 300))

# Using CSS selectors to pick out regions
webshot("http://rstudio.github.io/leaflet", "leaflet-menu.png", selector = ".list-group")
webshot("http://reddit.com/", "reddit-top.png",
        selector = c("input[type='text']", "#header-bottom-left"))

# Expand selection region
webshot("http://rstudio.github.io/leaflet", "leaflet-boxes.png",
        selector = "#installation", expand = c(10, 50, 0, 50))

# If multiple matches for a given selector, it uses the first match
webshot("http://rstudio.github.io/leaflet", "leaflet-p.png", selector = "p")
webshot("https://github.com/rstudio/shiny/", "shiny-stats.png",
         selector = "ul.numbers-summary")

# Send commands to eval
webshot("http://www.reddit.com/", "reddit-input.png",
  selector = c("#search", "#login_login-main"),
  eval = "casper.then(function() {
    // Check the remember me box
```

```
    this.click('#rem-login-main');
    // Enter username and password
    this.sendKeys('#login_login-main input[type=\"text\"]', 'my_username');
    this.sendKeys('#login_login-main input[type=\"password\"]', 'password');

    // Now click in the search box. This results in a box expanding below
    this.click('#search input[type=\"text\"]');
    // Wait 500ms
    this.wait(500);
  });"
)

# Result can be piped to other commands like resize() and shrink()
webshot("https://www.r-project.org/", "r-small.png") %>%
 resize("75%") %>%
 shrink()

# Requests can change the User-Agent header
webshot(
  "https://www.rstudio.com/products/rstudio/download/",
  "rstudio.png",
  useragent = "Mozilla/5.0 (Macintosh; Intel Mac OS X)"
)

# See more examples in the package vignette
}
```

Index

appshot, 2, *9*

install_phantomjs, 3
is_phantomjs_installed, 4

render, *6*
resize, 5
rmdshot, 5
run, *6*

shrink, 7

webshot, *2, 3*, *6*, 7