Package 'webshot'

August 29, 2016

Title Take Screenshots of Web Pages

Version 0.3.2

Description Takes screenshots of web pages, including Shiny applications.

Depends R (>= 3.0)

Imports magrittr

Suggests httpuv, knitr, rmarkdown, shiny

VignetteBuilder knitr

License GPL-2

SystemRequirements PhantomJS (http://phantomjs.org) for taking screenshots, ImageMagick (http://www.imagemagick.org) or GraphicsMagick (http://www.graphicsmagick.org) and OptiPNG (http://optipng.sourceforge.net) for manipulating images.

RoxygenNote 5.0.1

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

NeedsCompilation no

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

Maintainer Winston Chang <winston@rstudio.com>

Repository CRAN

Date/Publication 2016-06-23 00:47:07

R topics documented:

appshot 2
install_phantomjs 2
resize 3
shrink 4
webshot 5

Index **8**

appshot *Take a screenshot of a Shiny app*

Description

Take a screenshot of a Shiny app

Usage

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

Arguments

app	A Shiny app object, or a string naming an app directory.
file	Name of output file. Should end with .png, .pdf, or .jpeg.
...	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.

Examples

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

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/".

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.

Value

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

---

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	Name of image 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.

Examples

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

  # Generate image that is 400 pixels wide
  webshot("http://www.google.com/", "google-small-2.png") %>%
    resize("400x")
}
```

`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`	Name of image to shrink. Must be a PNG file.

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("http://www.google.com/", "google-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,
  eval = NULL)
```

Arguments

`url`	A URL to visit.
`file`	Name of output file. Should end with `.png`, `.pdf`, or `.jpeg`.
`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. 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`.
`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.
`delay`	Time to wait before taking screenshot, in seconds. Sometimes a longer delay is needed for all assets to display properly.
`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 (`http://docs.casperjs.org/en/latest/modules/casper.html`) for more information about what commands can be used to control the web page. NOTE: This is experimental and likely to change!

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)

# 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("http://www.google.com/", "google-small.png") %>%
 resize("75%") %>%
 shrink()
```

```
}

# See more examples in the package vignette
vignette("intro", package = "webshot")
```

Index

appshot, 2, *5*

install_phantomjs, 2

resize, 3

shrink, 4

webshot, *2*, 5