

Package ‘downloader’

July 2, 2014

Maintainer Winston Chang <winston@stdout.org>

Author Winston Chang <winston@stdout.org>

Version 0.3

License GPL-2

Title A package for downloading files over http and https

Description This package provides a wrapper for the `download.file` function, making it possible to download files over https on Windows, Mac OS X, and other Unix-like platforms. The `RCurl` package provides this functionality (and much more) but can be difficult to install because it must be compiled with external dependencies. This package has no external dependencies, so it is much easier to install.

URL <https://github.com/wch/downloader>

Imports digest

Suggests testthat

Collate 'download.r' 'downloader-package.r' 'source_url.r' 'sha_url.r'

NeedsCompilation no

Repository CRAN

Date/Publication 2013-05-01 07:23:57

R topics documented:

download	2
downloader	3
sha_url	3
source_url	4

Index	6
--------------	----------

download	<i>Download a file, using http, https, or ftp</i>
----------	---

Description

This is a wrapper for [download.file](#) and takes all the same arguments. The only difference is that, if the protocol is https, it changes some settings to make it work. How exactly the settings are changed differs among platforms.

Usage

```
download(url, ...)
```

Arguments

url	The URL to download.
...	Other arguments that are passed to download.file .

Details

This function also should follow http redirects on all platforms, which is something that does not happen by default when `curl` is used, as on Mac OS X.

With Windows, it calls `setInternet2`, which tells R to use the `internet2.dll`. Then it downloads the file by calling [download.file](#) using the "internal" method.

On other platforms, it will try to use `wget`, then `curl`, and then `lynx` to download the file. Typically, Linux platforms will have `wget` installed, and Mac OS X will have `curl`.

Note that for many (perhaps most) types of files, you will want to use `mode="wb"` so that the file is downloaded in binary mode.

See Also

[download.file](#) for more information on the arguments that can be used with this function.

Examples

```
## Not run:  
# Download the downloader source, in binary mode  
download("https://github.com/wch/downloader/zipball/master",  
         "downloader.zip", mode = "wb")  
  
## End(Not run)
```

downloader	<i>downloader: a package for making it easier to download files over https</i>
------------	--

Description

This package provides a wrapper for the `download.file` function, making it possible to download files over https on Windows, Mac OS X, and other Unix-like platforms. The `RCurl` package provides this functionality (and much more) but can be difficult to install because it must be compiled with external dependencies. This package has no external dependencies, so it is much easier to install.

sha_url	<i>Download a file from a URL and find a SHA-1 hash of it</i>
---------	---

Description

This will download a file and find a SHA-1 hash of it, using `digest()`. The primary purpose of this function is to provide an easy way to find the value of `sha` which can be passed to `source_url()`.

Usage

```
sha_url(url, cmd = TRUE)
```

Arguments

<code>url</code>	The URL of the file to find a hash of.
<code>cmd</code>	If TRUE (the default), print out a command for sourcing the URL with <code>source_url()</code> , including the hash.

Examples

```
## Not run:
# Get the SHA hash of a file. It will print the text below and return
# the hash as a string
sha_url("https://gist.github.com/wch/dae7c106ee99fe1fdfe7/raw/db0c9bfe0de85d15c60b0b9bf22403c0f5e1fb15/test.r")
# Command for sourcing the URL:
# downloader::source_url("https://gist.github.com/wch/dae7c106ee99fe1fdfe7/raw/db0c9bfe0de85d15c60b0b9bf22403c0f5e1fb15/test.r")
# [1] "9b8ff5213e32a871d6cb95cce0bed35c53307f61"

## End(Not run)
```

source_url	<i>Download an R file from a URL and source it</i>
------------	--

Description

This will download a file and source it. Because it uses the [download\(\)](#) function, it can handle https URLs.

Usage

```
source_url(url, sha = NULL, ..., prompt = TRUE,
           quiet = FALSE)
```

Arguments

url	The URL to download.
sha	A SHA-1 hash of the file at the URL.
prompt	Prompt the user if no value for sha is provided.
quiet	If FALSE (the default), print out status messages about checking SHA.
...	Other arguments that are passed to source() .

Details

By default, `source_url()` checks the SHA-1 hash of the file. If it differs from the expected value, it will throw an error. The default expectation is that a hash is provided; if not, `source_url()` will prompt the user, asking if they are sure they want to continue, unless `prompt=FALSE` is used. In other words, if you use `prompt=FALSE`, it will run the remote code without checking the hash, and without asking the user.

The purpose of checking the hash is to ensure that the file has not changed. If a `source_url` command with a hash is posted in a public forum, then others who source the URL (with the hash) are guaranteed to run the same code every time. This means that the author doesn't need to worry about the security of the server hosting the file. It also means that the users don't have to worry about the file being replaced with a damaged or maliciously-modified version.

To find the hash of a local file, use [digest\(\)](#). For a simple way to find the hash of a remote file, use [sha_url\(\)](#).

See Also

[source\(\)](#) for more information on the arguments that can be used with this function.

Examples

```
## Not run:  
# Source the a sample file  
downloader::source_url("https://gist.github.com/wch/dae7c106ee99fe1fdfe7/raw/db0c9bfe0de85d15c60b0b9bf22403c0f  
  sha="9b8ff5213e32a871d6cb95cce0bed35c53307f61")  
  
## End(Not run)
```

Index

digest, [3](#), [4](#)
download, [2](#), [4](#)
download.file, [2](#)
downloader, [3](#)
downloader-package (downloader), [3](#)

sha_url, [3](#), [4](#)
source, [4](#)
source_url, [3](#), [4](#)