Package 'Rcrawler'

November 11, 2018

Type Package

Title Web Crawler and Scraper

Version 0.1.9-1

Date 2018-11-11

Description Performs parallel web crawling and web scraping. It is designed to crawl, parse and store web pages to produce data that can be directly used for analysis application. For details see Khalil and Fakir (2017) <DOI:10.1016/j.softx.2017.04.004>.

License GPL (>= 2)

URL https://github.com/salimk/Rcrawler/

BugReports https://github.com/salimk/Rcrawler/issues

LazyData TRUE

Imports httr, xml2, data.table, foreach, doParallel, parallel, selectr, webdriver, callr, jsonlite

RoxygenNote 6.1.0

NeedsCompilation no

Author Salim Khalil [aut, cre] (<https://orcid.org/0000-0002-7804-4041>)

Maintainer Salim Khalil <khalilsalim1@gmail.com>

Repository CRAN

Date/Publication 2018-11-11 22:00:16 UTC

R topics documented:

`browser_path` *Return browser (webdriver) location path*

Description

After installing webdriver using `install_browser`, you can check its location path by running this function.

Usage

```
browser_path()
```

Value

path as character

Author(s)

salim khalil

Examples

```
## Not run:

browser_paths()

## End(Not run)
```

ContentScraper *ContentScraper*

Description

ContentScraper

Usage

```
ContentScraper(Url, HTmlText, browser, XpathPatterns, CssPatterns,
  PatternsName, ExcludeXpathPat, ExcludeCSSPat, ManyPerPattern = FALSE,
  astext = TRUE, asDataFrame = FALSE, encod)
```

Arguments

Url	character, one url or a vector of urls of web pages to scrape.
HTmlText	character, web page as HTML text to be scraped.use either Url or HtmlText not both.
browser,	a web driver session, or a loggedin session of the web driver (see examples)
XpathPatterns	character vector, one or more XPath patterns to extract from the web page.
CssPatterns	character vector, one or more CSS selector patterns to extract from the web page.
PatternsName	character vector, given names for each xpath pattern to extract, just as an indication .
ExcludeXpathPat	character vector, one or more Xpath pattern to exclude from extracted content (like excluding quotes from forum replies or excluding middle ads from Blog post) .
ExcludeCSSPat	character vector, one or more Css pattern to exclude from extracted content.
ManyPerPattern	boolean, If False only the first matched element by the pattern is extracted (like in Blogs one page has one article/post and one title). Otherwise if set to True all nodes matching the pattern are extracted (Like in galleries, listing or comments, one page has many elements with the same pattern)
astext	boolean, default is TRUE, HTML and PHP tags is stripped from the extracted piece.
asDataFrame	boolean, transform scraped data into a Dataframe. default is False (data is returned as List)
encod	character, set the weppage character encoding.

Value

return a named list of scraped content

Author(s)

salim khalil

Examples

```
## Not run:

#### Extract title, publishing date and article from the web page using css selectors
#
DATA<-ContentScraper(Url="http://glofile.com/index.php/2017/06/08/taux-nette-detente/",
CssPatterns = c(".entry-title",".published",".entry-content"), astext = TRUE)

#### The web page source can be provided also in HTML text (characters)
#
txthml<-"<html><title>blah</title><div><p>I m the content</p></div></html>"
DATA<-ContentScraper(HTmlText = txthml ,XpathPatterns = "//*/p")

#### Extract post title and bodt from the web page using Xpath patterns,
#  PatternsName can be provided as indication.
#
DATA<-ContentScraper(Url ="http://glofile.com/index.php/2017/06/08/athletisme-m-a-rome/",
XpathPatterns=c("//head/title","//*/article"),PatternsName=c("title", "article"))

#### Extract titles and contents of 3 Urls using CSS selectors, As result DATA variable
# will handle 6 elements.
#
urllist<-c("http://glofile.com/index.php/2017/06/08/sondage-quel-budget/",
"http://glofile.com/index.php/2017/06/08/cyril-hanouna-tire-a-boulets-rouges-sur-le-csa/",
"http://glofile.com/index.php/2017/06/08/placements-quelles-solutions-pour-doper/",
"http://glofile.com/index.php/2017/06/08/paris-un-concentre-de-suspens/")
DATA<-ContentScraper(Url =urllist, CssPatterns = c(".entry-title",".entry-content"),
PatternsName = c("title","content"))

#### Extract post title and list of comments from a set of blog pages,
# ManyPerPattern argument enables extracting many elements having same pattern from each
# page like comments, reviews, quotes and listing.
DATA<-ContentScraper(Url =urllist, CssPatterns = c(".entry-title",".comment-content p"),
PatternsName = c("title","comments"), astext = TRUE, ManyPerPattern = TRUE)
#### From this Forum page  e extract the post title and all replies using CSS selectors
# c("head > title",".post"), However, we know that each reply contain previous Replys
# as quote so we need to exclude To remove inner quotes in each reply we use
# ExcludeCSSPat c(".quote",".quoteheader a")
DATA<-ContentScraper(Url = "https://bitcointalk.org/index.php?topic=2334331.0",
CssPatterns = c("head > title",".post"), ExcludeCSSPat = c(".quote",".quoteheader"),
PatternsName = c("Title","Replys"), ManyPerPattern = TRUE)

#### Scrape data from web page requiring authentification
# replace \@ by @ before running follwing examples
# create a loggedin session
LS<-run_browser()
LS<-LoginSession(Browser = LS, LoginURL = 'https://manager.submittable.com/login',
   LoginCredentials = c('your email','your password'),
   cssLoginFields =c('#email', '#password'),
   XpathLoginButton ='//*[\@type=\"submit\"]' )
#Then scrape data with the session
DATA<-ContentScraper(Url='https://manager.submittable.com/beta/discover/119087',
```

```
     XpathPatterns = c('//*[\@id=\"submitter-app\"]/div/div[2]/div/div/div/div/div[3]',
      '//*[\@id=\"submitter-app\"]/div/div[2]/div/div/div/div/div[2]/div[1]/div[1]' ),
      PatternsName = c("Article","Title"), astext = TRUE, browser = LS )
#OR
page<-LinkExtractor(url='https://manager.submittable.com/beta/discover/119087',
                    browser = LS)
DATA<-ContentScraper(HTmlText = page$Info$Source_page,
     XpathPatterns = c("//*[\@id=\"submitter-app\"]/div/div[2]/div/div/div/div/div[3]",
     "//*[\@id=\"submitter-app\"]/div/div[2]/div/div/div/div/div[2]/div[1]/div[1]" ),
      PatternsName = c("Article","Title"),astext = TRUE )

To get all first elements of the lists in one vector (example all titles) :
VecTitle<-unlist(lapply(DATA, `[[`, 1))
To get all second elements of the lists in one vector (example all articles)
VecContent<-unlist(lapply(DATA, `[[`, 2))


## End(Not run)
```

`Drv_fetchpage` *Fetch page using web driver/Session*

Description

Fetch page using web driver/Session

Usage

```
Drv_fetchpage(url, browser)
```

Arguments

`url`	character, web page URL to retreive
`browser`	Object returned by `run_browser`

Value

return a list of three elements, the first is a list containing the web page details (url, encoding-type, content-type, content ... etc), the second is a character-vector containing the list of retreived internal urls and the third is a vetcor of external Urls.

Author(s)

salim khalil

Getencoding *Getencoding*

Description

This function retreives the encoding charset of web page based on HTML tags and HTTP header

Usage

```
Getencoding(url)
```

Arguments

url	character, the web page url.

Value

return the encoding charset as character

Author(s)

salim khalil

install_browser *Install PhantomJS webdriver*

Description

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

Usage

```
install_browser(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 **webdriver** 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).

LinkExtractor *LinkExtractor*

Description

Fetch and parse a document by URL, to extract page info, HTML source and links (internal/external). Fetching process can be done by HTTP GET request or through webdriver (phantomjs) which simulate a real browser rendering.

Usage

```
LinkExtractor(url, id, lev, IndexErrPages, Useragent, Timeout = 6,
  use_proxy = NULL, URLlenlimit = 255, urlExtfilter, urlregexfilter,
  encod, urlbotfiler, removeparams, removeAllparams = FALSE,
  ExternalLInks = FALSE, urlsZoneXpath = NULL, Browser,
  RenderingDelay = 0)
```

Arguments

url	character, url to fetch and parse.
id	numeric, an id to identify a specific web page in a website collection, it's auto-generated byauto-generated by Rcrawler function.
lev	numeric, the depth level of the web page, auto-generated by Rcrawler function.
IndexErrPages	character vector, http error code-statut that can be processed, by default, it's IndexErrPages<-c(200) which means only successfull page request should be parsed .Eg, To parse also 404 error pages add, IndexErrPages<-c(200,404).
Useragent	, the name the request sender, default to "Rcrawler". but we recommand using a regular browser user-agent to avoid being blocked by some server.
Timeout	,default to 5s

use_proxy,	object created by httr::use_proxy() function, if you want to use a proxy to retreive web page. (does not work with webdriver).
URLlenlimit	interger, Maximum URL length to process, default to 255 characters (Useful to avoid spider traps)
urlExtfilter	character vector, the list of file extensions to exclude from parsing, Actualy, only html pages are processed(parsed, scraped); To define your own lis use urlExtfilter<-c(ext1,ext2,ext3)
urlregexfilter	character vector, filter out extracted internal urls by one or more regular expression.
encod	character, web page character encoding
urlbotfiler	character vector , directories/files restricted by robot.txt
removeparams	character vector, list of url parameters to be removed form web page internal links.
removeAllparams	boolean, IF TRUE the list of scraped urls will have no parameters.
ExternalLInks	boolean, default FALSE, if set to TRUE external links also are returned.
urlsZoneXpath,	xpath pattern of the section from where links should be exclusively gathered/collected.
Browser	the client object of a remote headless web driver(virtual browser), created by br<-run_browser() function, or a logged-in browser session object, created by LoginSession, after installing web driver Agent install_browser(). see examples below.
RenderingDelay	the time required by a webpage to be fully rendred, in seconds.

Value

return a list of three elements, the first is a list containing the web page details (url, encoding-type, content-type, content ... etc), the second is a character-vector containing the list of retreived internal urls and the third is a vetcor of external Urls.

Author(s)

salim khalil

Examples

```
## Not run:

###### Fetch a URL using GET request :
#####################################################
##
## Very Fast, but can't fetch javascript rendred pages or sections

# fetch the page with default config, then returns page info and internal links

page<-LinkExtractor(url="http://www.glofile.com")
```

```
# this will return  alse external links

page<-LinkExtractor(url="http://www.glofile.com", ExternalLInks = TRUE)

# Specify Useragent to overcome bots blocking by some websites rules

page<-LinkExtractor(url="http://www.glofile.com", ExternalLInks = TRUE,
       Useragent = "Mozilla/5.0 (Windows NT 6.3; Win64; x64)",)

# By default, only HTTP succeeded page are parsed, therefore, to force
# parse error pages like 404 you need to specify IndexErrPages,

page<-LinkExtractor(url="http://www.glofile.com/404notfoundpage",
      ExternalLInks = TRUE, IndexErrPages = c(200,404))


#### Use GET request with a proxy
#
proxy<-httr::use_proxy("190.90.100.205",41000)
pageinfo<-LinkExtractor(url="http://glofile.com/index.php/2017/06/08/taux-nette-detente/",
use_proxy = proxy)

#' Note : use_proxy arguments can' not't be configured with webdriver

###### Fetch a URL using a web driver (virtual browser)
#########################################################
##
## Slow, because a headless browser called phantomjs will simulate
## a user session on a website. It's useful for web page having important
## javascript rendred sections such as menus.
## We recommend that you first try normal previous request, if the function
## returns a forbidden 403 status code or an empty/incomplete source code body,
## then try to set a normal useragent like
## Useragent = "Mozilla/5.0 (Windows NT 6.3; Win64; x64)",
## if you still have issue then you shoud try to set up a virtual browser.

#1 Download and install phantomjs headless browser
install_browser()

#2 start browser process (takes 30 seconds usualy)
br <-run_browser()

#3 call the function
page<-LinkExtractor(url="http://www.master-maroc.com", Browser = br,
      ExternalLInks = TRUE)

#4 dont forget to stop the browser at the end of all your work with it
stop_browser(br)

###### Fetch a web page that requires authentication
#########################################################
## In some case you may need to retreive content from a web page which
```

```
## requires authentication via a login page like private forums, platforms..
## In this case you need to run \link{LoginSession} function to establish a
## authenticated browser session; then use \link{LinkExtractor} to fetch
## the URL using the auhenticated session.
## In the example below we will try to fech a private blog post which
## require authentification .

If you retreive the page using regular function LinkExtractor or your browser
page<-LinkExtractor("http://glofile.com/index.php/2017/06/08/jcdecaux/")
The post is not visible because it's private.
Now we will try to login to access this post using folowing creditentials
username : demo and password : rc@pass@r

#1 Download and install phantomjs headless browser (skip if installed)
install_browser()

#2 start browser process
br <-run_browser()

#3 create auhenticated session
#  see \link{LoginSession} for more details

 LS<-LoginSession(Browser = br, LoginURL = 'http://glofile.com/wp-login.php',
                LoginCredentials = c('demo','rc@pass@r'),
                cssLoginFields =c('#user_login', '#user_pass'),
                cssLoginButton='#wp-submit' )

#check if login successful
LS$session$getTitle()
#Or
LS$session$getUrl()
#Or
LS$session$takeScreenshot(file = 'sc.png')

#3 Retreive the target private page using the logged-in session
page<-LinkExtractor(url='http://glofile.com/index.php/2017/06/08/jcdecaux/',Browser = LS)

#4 dont forget to stop the browser at the end of all your work with it
stop_browser(LS)


################### Returned Values ####################
########################################################

# Returned 'page' variable should include :
# 1- list of page details,
# 2- Internal links
# 3- external links.

#1 Vector of extracted internal links  (in-links)
page$InternalLinks

#2 Vector of extracted external links  (out-links)
```

```
page$ExternalLinks

page$Info

# Requested Url
page$Info$Url

# Sum of extracted links
page$Info$SumLinks

# The status code of the HTTP response 200, 401, 300...
page$Info$Status_code

# The MIME type of this content from HTTP response
page$Info$Content_type

# Page text encoding UTF8, ISO-8859-1 , ..
page$Info$Encoding

# Page source code
page$Info$Source_page

Page title
page$Info$Title

Other returned values page$Info$Id, page$Info$Crawl_level,
page$Info$Crawl_status are only used by Rcrawler funtion.



## End(Not run)
```

`LinkNormalization` *Link Normalization*

Description

To normalize and transform URLs into a canonical form.

Usage

```
LinkNormalization(links, current)
```

Arguments

`links`	character, one or more URLs to Normalize.
`current`	character, The current page URL where links are located

Value

Vector of normalized urls

Author(s)

salim khalil

Examples

```
# Normalize a set of links

links<-c("http://www.twitter.com/share?url=http://glofile.com/page.html",
         "/finance/banks/page-2017.html",
         "./section/subscription.php",
         "//section/",
         "www.glofile.com/home/",
         "IndexEn.aspx",
         "glofile.com/sport/foot/page.html",
         "sub.glofile.com/index.php",
         "http://glofile.com/page.html#1",
         "?tags%5B%5D=votingrights&amp;sort=popular"
                  )

links<-LinkNormalization(links,"http://glofile.com" )

links
```

Linkparameters *Get the list of parameters and values from an URL*

Description

A function that take a URL _charachter_ as input, and extract the parameters and values from this URL .

Usage

```
Linkparameters(URL)
```

Arguments

URL	character, the URL to extract

Details

This function extract the link parameters and values (Up to 10 parameters)

Value

return the URL paremeters=values

Author(s)

salim khalil

Examples

```
Linkparameters("http://www.glogile.com/index.php?name=jake&age=23&template=2&filter=true")
# Extract all URL parameters with values as vector
```

Linkparamsfilter *Link parameters filter*

Description

This function remove a given set of parameters from a specific URL

Usage

```
Linkparamsfilter(URL, params, removeAllparams = FALSE)
```

Arguments

URL	character, the URL from which params and values have to be removed
params	character vector, List of url parameters to be removed
removeAllparams,	boolean if true , all url parameters will be removed.

Details

This function exclude given parameters from the urls,

Value

return a URL wihtout given parameters

Author(s)

salim khalil

Examples

```
#remove ord and tmp parameters from the URL
url<-"http://www.glogile.com/index.php?name=jake&age=23&tmp=2&ord=1"
url<-Linkparamsfilter(url,c("ord","tmp"))
#remove all URL parameters
Linkparamsfilter(url,removeAllparams = TRUE)
```

`ListProjects` *ListProjects*

Description

List all crawling project in your R local directory, or in a custom directory

Usage

```
ListProjects(DIR)
```

Arguments

`DIR`	character By default it's your local R workspace, if you set a custom folder for your crawling project then user DIR param to access this folder.

Value

`ListProjects`, a character vector.

Author(s)

salim khalil

Examples

```
## Not run:
ListProjects()

## End(Not run)
```

LoadHTMLFiles *LoadHTMLFiles @rdname LoadHTMLFiles*

Description

LoadHTMLFiles @rdname LoadHTMLFiles

Usage

```
LoadHTMLFiles(ProjectName, type = "vector", max)
```

Arguments

ProjectName	character, the name of the folder holding collected HTML files, use ListProjects fnuction to see all projects.
type	character, the type of returned variable, either vector or list.
max	Integer, maximum number of files to load.

Value

LoadHTMLFiles, a character vector or a list;

Author(s)

salim khalil

Examples

```
## Not run:
ListProjects()
#show all crawling project folders stored in your local R wokspace folder
DataHTML<-LoadHTMLFiles("glofile.com-301010")
#Load all HTML files in DataHTML vector
DataHTML2<-LoadHTMLFiles("glofile.com-301010",max = 10, type = "list")
#Load only 10 first HTMl files in DataHTML2 list

## End(Not run)
```

`LoginSession` *Open a logged in Session*

Description

Simulate authentifaction using web driver automation This function Fetch login page using phantomjs web driver(virtual browser), sets login and password values + other required values then clicks on login button. You should provide these agruments for the function to work correctly : - Login page URL - Login Credentials eg: email & password - css Or Xpath of Login Credential fields - css or xpath of Login Button - If a checkbox is required in the login page then provide provide its css or xpath pattern

Usage

```
LoginSession(Browser, LoginURL, LoginCredentials, cssLoginFields,
  cssLoginButton, cssRadioToCheck, XpathLoginFields, XpathLoginButton,
  XpathRadioToCheck)
```

Arguments

`Browser`	object, phatomjs web driver use `run_browser` function to create this object.
`LoginURL`	character, login page URL
`LoginCredentials,`	login Credentials values eg: email and password
`cssLoginFields,`	vector of login fields css pattern.
`cssLoginButton,`	the css pattern of the login button that should be clicked to access protected zone.
`cssRadioToCheck,`	the radio/checkbox css pattern to be checked(if exist)
`XpathLoginFields,`	vector of login fields xpath pattern.
`XpathLoginButton,`	the xpath pattern of the login button.
`XpathRadioToCheck`	the radio/checkbox xpath pattern to be checked(if exist)

Value

return authentified browser session object

Author(s)

salim khalil

Examples

```
## Not run:


 #This function is based on web browser automation, so, before start,
 make sure you have successfully installed web driver (phantomjs).
 install_browser()
 # Run browser process and get its reference object
 br<- run_browser()

  brs<-LoginSession(Browser = br, LoginURL = 'http://glofile.com/wp-login.php',
                LoginCredentials = c('demo','rc@pass@r'),
                cssLoginFields =c('#user_login', '#user_pass'),
                cssLoginButton='#wp-submit' )

 # To make sure that you have been successfully authenticated
 # Check URL of the current page after login redirection
 brs$getUrl()
 # Or Take screenshot of the website dashborad
 brs$takeScreenshot(file = "sc.png")


 brs$delete()
 brs$status()
 brs$go(url)
 brs$getUrl()
 brs$goBack()
 brs$goForward()
 brs$refresh()
 brs$getTitle()
 brs$getSource()
 brs$takeScreenshot(file = NULL)
 brs$findElement(css = NULL, linkText = NULL,
              partialLinkText = NULL, xpath = NULL)
 brs$findElements(css = NULL, linkText = NULL,
               partialLinkText = NULL, xpath = NULL)
 brs$executeScript(script, ...)
 brs$executeScriptAsync(script, ...)
 brs$setTimeout(script = NULL, pageLoad = NULL, implicit = NULL)
 brs$moveMouseTo(xoffset = 0, yoffset = 0)
 brs$click(button = c("left", "middle", "right"))
 brs$doubleClick(button = c("left", "middle", "right"))
 brs$mouseButtonDown(button = c("left", "middle", "right"))
 brs$mouseButtonUp(button = c("left", "middle", "right"))
 brs$readLog(type = c("browser", "har"))
 brs$getLogTypes()


## End(Not run)
```

Rcrawler *Rcrawler*

Description

The crawler's main function, by providing only the website URL and the Xpath or CSS selector patterns this function can crawl the whole website (traverse all web pages) download webpages, and scrape/extract its contents in an automated manner to produce a structured dataset. The process of a crawling operation is performed by several concurrent processes or nodes in parallel, so it's recommended to use 64bit version of R.

Usage

```
Rcrawler(Website, no_cores, no_conn, MaxDepth, DIR, RequestsDelay = 0,
  Obeyrobots = FALSE, Useragent, use_proxy = NULL, Encod,
  Timeout = 5, URLlenlimit = 255, urlExtfilter, dataUrlfilter,
  crawlUrlfilter, crawlZoneCSSPat = NULL, crawlZoneXPath = NULL,
  ignoreUrlParams, ignoreAllUrlParams = FALSE, KeywordsFilter,
  KeywordsAccuracy, FUNPageFilter, ExtractXpathPat, ExtractCSSPat,
  PatternsNames, ExcludeXpathPat, ExcludeCSSPat, ExtractAsText = TRUE,
  ManyPerPattern = FALSE, saveOnDisk = TRUE, NetworkData = FALSE,
  NetwExtLinks = FALSE, statslinks = FALSE, Vbrowser = FALSE,
  LoggedSession)
```

Arguments

Website	character, the root URL of the website to crawl and scrape.
no_cores	integer, specify the number of clusters (logical cpu) for parallel crawling, by default it's the numbers of available cores.
no_conn	integer, it's the number of concurrent connections per one core, by default it takes the same value of no_cores.
MaxDepth	integer, repsents the max deph level for the crawler, this is not the file depth in a directory structure, but 1+ number of links between this document and root document, default to 10.
DIR	character, correspond to the path of the local repository where all crawled data will be stored ex, "C:/collection" , by default R working directory.
RequestsDelay	integer, The time interval between each round of parallel http requests, in seconds used to avoid overload the website server. default to 0.
Obeyrobots	boolean, if TRUE, the crawler will parse the website\'s robots.txt file and obey its rules allowed and disallowed directories.
Useragent	character, the User-Agent HTTP header that is supplied with any HTTP requests made by this function.it is important to simulate different browser's user-agent to continue crawling without getting banned.
use_proxy	object created by httr::use_proxy() function, if you want to use a proxy (does not work with webdriver).

`Encod` character, set the website caharacter encoding, by default the crawler will automatically detect the website defined character encoding.

`Timeout` integer, the maximum request time, the number of seconds to wait for a response until giving up, in order to prevent wasting time waiting for responses from slow servers or huge pages, default to 5 sec.

`URLlenlimit` integer, the maximum URL length limit to crawl, to avoid spider traps; default to 255.

`urlExtfilter` character's vector, by default the crawler avoid irrelevant files for data scraping such us xml,js,css,pdf,zip ...etc, it's not recommanded to change the default value until you can provide all the list of filetypes to be escaped.

`dataUrlfilter` character's vector, filter Urls to be scraped/collected by one or more regular expression patterns.Useful to control which pages should be collected/scraped, like product, post, detail or category pages if they have a commun URL pattern. without start ^ and end $ regex.

`crawlUrlfilter` character's vector, filter Urls to be crawled by one or more regular expression patterns. Useful for large websites to control the crawler behaviour and which URLs should be crawled. For example, In case you want to crawl a website's search resutls (guided/oriented crawling). without start ^ and end $ regex.

`crawlZoneCSSPat`
one or more css pattern of page sections from where the crawler should gather links to be followed, to avoid navigating through all visible links and to have more control over the crawler behaviour in target website.

`crawlZoneXPath` one or more xpath pattern of page sections from where the crawler should gather links to be followed.

`ignoreUrlParams`
character's vector, the list of Url paremeter to be ignored during crawling. Some URL parameters are ony related to template view if not ignored will cause duplicate page (many web pages having the same content but have different URLs) .

`ignoreAllUrlParams,`
boolean, choose to ignore all Url parameter after "?" (Not recommended for Non-SEF CMS websites because only the index.php will be crawled)

`KeywordsFilter` character vector, For users who desires to scrape or collect only web pages that contains some keywords one or more. Rcrawler calculate an accuracy score based of the number of founded keywords. This parameter must be a vector with at least one keyword like c("mykeyword").

`KeywordsAccuracy`
integer value range bewteen 0 and 100, used only with KeywordsFilter parameter to determine the accuracy of web pages to collect. The web page Accuracy value is calculated using the number of matched keywords and their occurence.

`FUNPageFilter` function, filter out pages to be collected/scraped by a custom function (conditions, prediction, calssification model). This function should take a LinkExtractor object as arument then finally returns TRUE or FALSE.

`ExtractXpathPat`
character's vector, vector of xpath patterns to match for data extraction process.

ExtractCSSPat	character's vector, vector of CSS selector pattern to match for data extraction process.
PatternsNames	character vector, given names for each xpath pattern to extract.
ExcludeXpathPat	character's vector, one or more Xpath pattern to exclude from extracted content ExtractCSSPat or ExtractXpathPat (like excluding quotes from forum replies or excluding middle ads from Blog post) .
ExcludeCSSPat	character's vector, similar to ExcludeXpathPat but using Css selectors.
ExtractAsText	boolean, default is TRUE, HTML and PHP tags is stripped from the extracted piece.
ManyPerPattern	boolean, ManyPerPattern boolean, If False only the first matched element by the pattern is extracted (like in Blogs one page has one article/post and one title). Otherwise if set to True all nodes matching the pattern are extracted (Like in galleries, listing or comments, one page has many elements with the same pattern)
saveOnDisk	boolean, By default is true, the crawler will store crawled Html pages and extracted data CSV file on a specific folder. On the other hand you may wish to have DATA only in memory.
NetworkData	boolean, If set to TRUE, then the crawler map all the internal hyperlink connections within the given website and return DATA for Network construction using igraph or other tools.(two global variables is returned see details)
NetwExtLinks	boolean, If TRUE external hyperlinks (outlinks) also will be counted on Network edges and nodes.
statslinks	boolean, if TRUE, the crawler counts the number of input and output links of each crawled web page.
Vbrowser	boolean, If TRUE the crawler will use web driver phantomsjs (virtual browser) to fetch and parse web pages instead of GET request
LoggedSession	A loggedin browser session object, created by LoginSession function

Details

To start Rcrawler task you need to provide the root URL of the website you want to scrape, it could be a domain, a subdomain or a website section (eg. http://www.domain.com, http://sub.domain.com or http://www.domain.com/section/). The crawler then will retreive the web page and go through all its internal links. The crawler continue to follow and parse all website's links automatically on the site until all website's pages have been parsed.

The process of a crawling is performed by several concurrent processes or nodes in parallel, So, It is recommended to use R 64-bit version.

For more tutorials check https://github.com/salimk/Rcrawler/

To scrape content with complex character such as Arabic or Chinese, you need to run Sys.setlocale function then set the appropriate encoding in Rcrawler function.

If you want to learn more about web scraper/crawler architecture, functional properties and implementation using R language, Follow this link and download the published paper for free .

Link: http://www.sciencedirect.com/science/article/pii/S2352711017300110

Dont forget to cite Rcrawler paper:

Khalil, S., & Fakir, M. (2017). RCrawler: An R package for parallel web crawling and scraping. SoftwareX, 6, 98-106.

Value

The crawling and scraping process may take a long time to finish, therefore, to avoid data loss in the case that a function crashes or stopped in the middle of action, some important data are exported at every iteration to R global environement:

- INDEX: A data frame in global environement representing the generic URL index,including the list of fetched URLs and page details (contenttype,HTTP state, number of out-links and in-links, encoding type, and level).

- A repository in workspace that contains all downloaded pages (.html files)

Data scraping is enabled by setting ExtractXpathPat or ExtractCSSPat parameter:

- DATA: A list of lists in global environement holding scraped contents.

- A csv file 'extracted_contents.csv' holding all extracted data.

If NetworkData is set to TRUE two additional global variables returned by the function are:

- NetwIndex : Vector maps alls hyperlinks (nodes) with a unique integer ID

- NetwEdges : data.frame representing edges of the network, with these column : From, To, Weight (the Depth level where the link connection has been discovered) and Type (1 for internal hyperlinks 2 for external hyperlinks).

Author(s)

salim khalil

Examples

```
## Not run:

######### Crawl, index, and store all pages of a websites using 4 cores and 4 parallel requests
#
Rcrawler(Website ="http://glofile.com/", no_cores = 4, no_conn = 4)

######### Crawl and index the website using 8 cores and 8 parallel requests with respect to
# robot.txt rules using Mozilla string in user agent.

Rcrawler(Website = "http://www.example.com/", no_cores=8, no_conn=8, Obeyrobots = TRUE,
Useragent="Mozilla 3.11")

######### Crawl the website using the default configuration and scrape specific data from
# the website, in this case we need all posts (articles and titles) matching two XPath patterns.
# we know that all blog posts have datesin their URLs like 2017/09/08 so to avoid
# collecting category or other pages we can tell the crawler that desired page's URLs
# are like 4-digit/2-digit/2-digit/ using regular expression.
# Note thatyou can use the excludepattern  parameter to exclude a node from being
# extracted, e.g., in the case that a desired node includes (is a parent of) an
```

```
# undesired "child" node. (article having inner ads or menu)

Rcrawler(Website = "http://www.glofile.com/", dataUrlfilter =  "/[0-9]{4}/[0-9]{2}/",
ExtractXpathPat = c("//*/article","//*/h1"), PatternsNames = c("content","title"))

######### Crawl the website. and collect pages having URLs matching this regular expression
# pattern (/[0-9]{4}/[0-9]{2}/). Collected pages will be stored in a local repository
# named "myrepo". And The crawler stops After reaching the third level of website depth.

 Rcrawler(Website = "http://www.example.com/", no_cores = 4, no_conn = 4,
 dataUrlfilter =  "/[0-9]{4}/[0-9]{2}/", DIR = "./myrepo", MaxDepth=3)


######### Crawl the website and collect/scrape only webpage related to a topic
# Crawl the website and collect pages containing keyword1 or keyword2 or both.
# To crawl a website and collect/scrape only some web pages related to a specific topic,
# like gathering posts related to Donald trump from a news website. Rcrawler function
# has two useful parameters KeywordsFilter and KeywordsAccuracy.
#
# KeywordsFilter : a character vector, here you should provide keywords/terms of the topic
# you are looking for. Rcrawler will calculate an accuracy score based on matched keywords
# and their occurrence on the page, then it collects or scrapes only web pages with at
# least a score of 1% wich mean at least one keyword is founded one time on the page.
# This parameter must be a vector with at least one keyword like c("mykeyword").
#
# KeywordsAccuracy: Integer value range between 0 and 100, used only in combination with
# KeywordsFilter parameter to determine the minimum accuracy of web pages to be collected
# /scraped. You can use one or more search terms; the accuracy will be calculated based on
# how many provided keywords are found on on the page plus their occurrence rate.
# For example, if only one keyword is provided c("keyword"), 50% means one occurrence of
# "keyword" in the page 100% means five occurrences of "keyword" in the page

 Rcrawler(Website = "http://www.example.com/", KeywordsFilter = c("keyword1", "keyword2"))

# Crawl the website and collect webpages that has an accuracy percentage higher than 50%
# of matching keyword1 and keyword2.

 Rcrawler(Website = "http://www.example.com/", KeywordsFilter = c("keyword1", "keyword2"),
   KeywordsAccuracy = 50)


######### Crawl a website search results
# In the case of scraping web pages specific to a topic of your interest; The methods
# above has some disadvantages which are complexity and time consuming as the whole
# website need to be crawled and each page is analyzed to findout desired pages.
# As result you may want to make use of the search box of the website and then directly
# crawl only search result pages. To do so, you may use \code{crawlUrlfilter} and
# \code{dataUrlfilter} arguments or \code{crawlZoneCSSPat}/\code{CrawlZoneXPath} with
\code{dataUrlfilter}.
#- \code{crawlUrlfilter}:what urls shoud be crawled (followed).
#- \code{dataUrlfilter}: what urls should be collected (download HTML or extract data ).
#- \code{crawlZoneCSSPat} Or \code{CrawlZoneXPath}: the page section where links to be
    crawled are located.
```

```
# Example1
# the command below will crawl all result pages knowing that result pages are like :
   http://glofile.com/?s=sur
   http://glofile.com/page/2/?s=sur
   http://glofile.com/page/2/?s=sur
# so they all have "s=sur" in common
# Post pages should be crawled also, post urls are like
  http://glofile.com/2017/06/08/placements-quelles-solutions-pour-dper/
  http://glofile.com/2017/06/08/taux-nette-detente/
# which contain a date format march regex "[0-9]{4}/[0-9]{2}/[0-9]{2}

Rcrawler(Website = "http://glofile.com/?s=sur", no_cores = 4, no_conn = 4,
crawlUrlfilter = c("[0-9]{4}/[0-9]{2}/[0-9]d{2}","s=sur"))

# In addition by using dataUrlfilter we specify that :
#  1- only post pages should be collected/scraped not all crawled result pages
#  2- additional urls should not be retreived from post page
#  (like post urls listed in 'related topic' or 'see more' sections)

Rcrawler(Website = "http://glofile.com/?s=sur", no_cores = 4, no_conn = 4,
crawlUrlfilter = c("[0-9]{4}/[0-9]{2}/[0-9]d{2}","s=sur"),
dataUrlfilter = "[0-9]{4}/[0-9]{2}/[0-9]{2}")

# Example 2
# collect job pages from indeed search result of "data analyst"

Rcrawler(Website = "https://www.indeed.com/jobs?q=data+analyst&l=Tampa,+FL",
 no_cores = 4 , no_conn = 4,
 crawlUrlfilter = c("/rc/","start="), dataUrlfilter = "/rc/")
# To include related post jobs on each collected post remove dataUrlfilter

# Example 3
# One other way to control the crawler behaviour, and to avoid fetching
# unnecessary links is to indicate to crawler the page zone of interest
# (a page section from where links should be grabed and crawled).
# The follwing example is similar to the last one,except this time we provide
# the xpath pattern of results search section to be crawled with all links within.

Rcrawler(Website = "https://www.indeed.com/jobs?q=data+analyst&l=Tampa,+FL",
 no_cores = 4 , no_conn = 4,MaxDepth = 3,
 crawlZoneXPath = c("//*[\@id='resultsCol']"), dataUrlfilter = "/rc/")


######### crawl and scrape a forum posts and replays, each page has a title and
# a list of replays , ExtractCSSPat = c("head>title","div[class=\"post\"]") .
# All replays have the same pattern, therfore we set TRUE ManyPerPattern
# to extract all of them.

Rcrawler(Website = "https://bitcointalk.org/", ManyPerPattern = TRUE,
ExtractCSSPat = c("head>title","div[class=\"post\"]"),
no_cores = 4, no_conn =4, PatternsName = c("Title","Replays"))
```

```
######### scrape data/collect pages meeting your custom criteria,
# This is useful when filetring by keyword or urls does not fullfil your needs, for example
# if you want to detect target pages  by classification/prediction model, or simply by checking
# a sppecifi text value/field in the web page, you can create a custom filter function for
# page selection as follow.
# First will create and test our function and test it with un one page .

pageinfo<-LinkExtractor(url="http://glofile.com/index.php/2017/06/08/sondage-quel-budget/",
encod=encod, ExternalLInks = TRUE)

Customfilterfunc<-function(pageinfo){
 decision<-FALSE
 # put your conditions here
   if(pageinfo$Info$Source_page ... ) ....
 # then return a boolean value TRUE : should be collected / FALSE should be escaped

 return TRUE or FALSE
}
 # Finally, you just call it inside Rcrawler function, Then the crawler will evaluate each
  page using your set of rules.

Rcrawler(Website = "http://glofile.com", no_cores=2, FUNPageFilter= Customfilterfunc )

######### Website Network
# Crawl the entire website, and create network edges DATA of internal links.
# Using Igraph for exmaple you can plot the network by the following commands

  Rcrawler(Website = "http://glofile.com/" , no_cores = 4, no_conn = 4, NetworkData = TRUE)
   library(igraph)
   network<-graph.data.frame(NetwEdges, directed=T)
   plot(network)

 # Crawl the entire website, and create network edges DATA of internal and external links .
 Rcrawler(Website = "http://glofile.com/" , no_cores = 4, no_conn = 4, NetworkData = TRUE,
  NetwExtLinks = TRUE)

###### Crawl a website using a web driver (Vitural browser)
##########################################################################
## In some case you may need to retreive content from a web page which
## requires authentication via a login page like private forums, platforms..
## In this case you need to run \link{LoginSession} function to establish a
## authenticated browser session; then use \link{LinkExtractor} to fetch
## the URL using the auhenticated session.
## In the example below we will try to fech a private blog post which
## require authentification .

If you retreive the page using regular function LinkExtractor or your browser
page<-LinkExtractor("http://glofile.com/index.php/2017/06/08/jcdecaux/")
The post is not visible because it's private.
Now we will try to login to access this post using folowing creditentials
username : demo and password : rc@pass@r
```

```
#1 Download and install phantomjs headless browser (skip if installed)
install_browser()

#2 start browser process
br <-run_browser()

#3 create auhenticated session
#  see \link{LoginSession} for more details

 LS<-LoginSession(Browser = br, LoginURL = 'http://glofile.com/wp-login.php',
                LoginCredentials = c('demo','rc@pass@r'),
                cssLoginCredentials =c('#user_login', '#user_pass'),
                cssLoginButton='#wp-submit' )

#check if login successful
LS$session$getTitle()
#Or
LS$session$getUrl()
#Or
LS$session$takeScreenshot(file = 'sc.png')
LS$session$getUrl()
LS<-run_browser()
LS<-LoginSession(Browser = LS, LoginURL = 'https://manager.submittable.com/login',
   LoginCredentials = c('your email','your password'),
   cssLoginFields =c('#email', '#password'),
   XpathLoginButton ='//*[\@type=\"submit\"]' )


# page<-LinkExtractor(url='https://manager.submittable.com/beta/discover/119087',
LoggedSession = LS)
# cont<-ContentScraper(HTmlText = page$Info$Source_page,
XpathPatterns = c("//*[\@id=\"submitter-app\"]/div/div[2]/div/div/div/div/div[3]",
"//*[\@id=\"submitter-app\"]/div/div[2]/div/div/div/div/div[2]/div[1]/div[1]" ),
PatternsName = c("Article","Title"),astext = TRUE )


## End(Not run)
```

RobotParser *RobotParser fetch and parse robots.txt*

Description

This function fetch and parse robots.txt file of the website which is specified in the first argument and return the list of corresponding rules .

Usage

```
RobotParser(website, useragent)
```

Arguments

website	character, url of the website which rules have to be extracted .
useragent	character, the useragent of the crawler

Value

return a list of three elements, the first is a character vector of Disallowed directories, the third is a Boolean value which is TRUE if the user agent of the crawler is blocked.

Examples

```
#RobotParser("http://www.glofile.com","AgentX")
#Return robot.txt rules and check whether AgentX is blocked or not.
```

---

run_browser *Start up web driver process on localhost, with a random port*

Description

Phantomjs is a headless browser, it provide automated control of a web page in an environment similar to web browsers, but via a command-line. It's able to render and understand HTML the same way a regular browser would, including styling elements such as page layout, colors, font selection and execution of JavaScript and AJAX which are usually not available when using GET request methods.

Usage

```
run_browser(debugLevel = "DEBUG", timeout = 5000)
```

Arguments

debugLevel	debug level, possible values: 'INFO', 'ERROR', 'WARN', 'DEBUG'
timeout	How long to wait (in milliseconds) for the webdriver connection to be established to the phantomjs process.

Details

This function will throw an error if webdriver(phantomjs) cannot be found, or cannot be started. It works with a timeout of five seconds.

If you got the forllwing error, this means that your operating system or antivirus is bloking the webdriver (phantom.js) process, try to disable your antivirus temporarily or adjust your system configuration to allow phantomjs and processx executable (browser_path to know where phantomjs is located). Error in supervisor_start() : processx supervisor was not ready after 5 seconds.

Value

A list of `callr::process` object, and `port`, the local port where phantom is running.

Examples

```
## Not run:

#If driver is not installed yet then
install_browser()

br<-run_browser()


## End(Not run)
```

`stop_browser` *Stop web driver process and Remove its Object*

Description

At the end of All your operations with the web river, you should stop its process and remove the driver R object else you may have troubles restarting R normaly. Throws and error if webdriver phantomjs cannot be found, or cannot be started. It works with a timeout of five seconds.

Usage

```
stop_browser(browser)
```

Arguments

`browser`	the web driver object created by `run_browser`

Value

A list of `process`, the `callr::process` object, and `port`, the local port where phantom is running.

Examples

```
## Not run:

#Start the browser
br<-run_browser()

#kill the browser process
stop_browser(br)
#remove the object reference
rm(br)
```

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

Index