Package: httpRequest (via r-universe)

October 10, 2024

Version 0.0.11

Date 2022-04-25

Title Basic HTTP Request

Author Eryk Witold Wolski, Andreas Westfeld

Maintainer Andreas Westfeld <andreas.westfeld@htw-dresden.de>

Description HTTP Request protocols. Implements the GET, POST and multipart POST request.

License GPL (>= 2)

NeedsCompilation no

Date/Publication 2022-04-25 06:40:08 UTC

Repository https://aw256.r-universe.dev

RemoteUrl https://github.com/cran/httpRequest

RemoteRef HEAD

RemoteSha 73df3cbf31aca1d4c734d35add35768ea746eef2

Contents

getToHost 2
getToHost2 3
postToHost 4
simplePostToHost 5

Index **7**

getToHost *Sends Get Request to HTTP Server (host)*

Description

Sends Get Request to HTTP Server

Usage

```
getToHost(host,path,referer,port=80)
```

Arguments

`host`	The host to which to send the Request e. g.: www.spiegel.de, 127.0.0.1
`path`	The path to the file eg. /cgi/getpasswords.pl?test=best or /index.html
`referer`	something like www.myhome.org
`port`	its 80 or 8080 most frequently

Details

It is a simple http client. So it does not take care of special chars e.g. whitspaces. For details see e.g. perl HTTP::Request::Common documentation and
`http://www.w3.org/Protocols/rfc1341/0_TableOfContents.html`

Value

The document which the server returns as a string.

Author(s)

E.W.Wolski wolski@molgen.mpg.de

See Also

postToHost, simplePostToHost

Examples

```
#check first if www.molgen.mpg.de is running.
#port <- 80
#getToHost("www.molgen.mpg.de",
#        "/~wolski/test.php4?test=test1&test2=test2&test3=3",
#        "www.test.pl", port=port)
```

getToHost2 *Sends Get Request to HTTP Server (host) with raw contents*

Description

Sends Get Request to HTTP Server (experimental version of getToHost based on raw data type)

Usage

```
getToHost2(host,path,referer,port=80)
```

Arguments

`host`	The host to which to send the Request e. g.: www.spiegel.de, 127.0.0.1
`path`	The path to the file eg. /cgi/getpasswords.pl?test=best or /index.html
`referer`	something like www.myhome.org
`port`	its 80 or 8080 most frequently

Details

It is a simple http client. So it does not take care of special chars e.g. whitspaces. For details see e.g. perl HTTP::Request::Common documentation and
`http://www.w3.org/Protocols/rfc1341/0_TableOfContents.html`

Value

The document which the server returns as a string.

Author(s)

E.W.Wolski wolski@molgen.mpg.de

See Also

getToHost

Examples

```
#check first if www.molgen.mpg.de is running.
#port <- 80
#getToHost2("www.molgen.mpg.de",
#        "/~wolski/test.php4?test=test1&test2=test2&test3=3",
#        "www.test.pl", port=port)
```

postToHost *Sends Multiform Post Request to HTTP Server*

Description

Sends Multiform Post Request to HTTP Server

Usage

```
postToHost(host, path, data.to.send, referer, port=80, ua, accept,
accept.language, accept.encoding, accept.charset, contenttype, cookie)
```

Arguments

`host`	The host to which to send the Request e. g.: www.spiegel.de, 127.0.0.1
`path`	The path to the file e. g. /cgi/getpasswords.pl or /index.html, default "/"
`data.to.send`	is a list of name value pairs e. g. list(name=value, name1=value1, name2=value2). The value may be a list("filename"=X, "object"=Y), where Y is a vector of type raw and X is a name to be posted as the filename of Y.
`referer`	something like www.myhome.org, default: NULL (not present in header)
`port`	port to connect to, default 80
`ua`	Identifier of the user agent, default "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.5) Gecko/20070719 Iceweasel/2.0.0.5 (Debian-2.0.0.5-2)"
`accept`	document types that are accepted by the client, default: "text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8, image/png,*/*;q=0.5"
`accept.language`	languages that are preferred by the client, default: "de,de-de;q=0.8,en-us;q=0.5,en;q=0.3"
`accept.encoding`	compressions that are accepted by the client, default: "gzip,deflate"
`accept.charset`	charset accepted by the client, default: "ISO-8859-1,utf-8;q=0.7,*;q=0.7"
`contenttype`	type of data that is sent to the server, default: "application/octet-stream"
`cookie`	e. g., "SessionID=1008XVG0F1F5D8H94294967295", default: NULL (not present in header)

Details

It is a simple http client. So it does not take care of special chars e.g. whitspaces. For details see e.g. perl HTTP::Request::Common documentation and
`http://www.w3.org/Protocols/rfc1341/0_TableOfContents.html`

Value

document which the server returns.

Author(s)

E.W.Wolski wolski@molgen.mpg.de, Andreas Westfeld andreas.westfeld@htw-dresden.de

See Also

getToHost, simplePostToHost

Examples

```
#to test uncomment. First check that the host is running.
#port <- 80
#test2 <- list(
# "fruit"="apple",
# "dat_defs"="20021204/F113213.dat",
# "myimage"=list(
#   "filename"="myimage.raw",
#   "object"=as.raw(as.vector(mymatrix))
#  ),
# "upsa"="test"
#)
#postToHost("www.molgen.mpg.de", "/~wolski/test.php4", test2,
#        referer="www.test.de", port=port)
```

`simplePostToHost` *Sends Simple Post Request to HTTP Server (host)*

Description

Sends Post Request to HTTP Server

Usage

```
simplePostToHost(host, path, datatosend, referer, contenttype,
                          port=80, maxlen=131063L)
```

Arguments

`host`	The host to which to send the Request eg.: www.spiegel.de, localhost
`path`	The path to the file eg. '`/cgi-bin/paramlist.cgi`' or '`/index.html`'
`datatosend`	key value pairs pairs separated by \& `"pid=14\&pollvotenumber=2\n4"`
`referer`	something like www.myhome.org
`contenttype`	default: application/x-www-form-urlencoded
`port`	its 80 or 8080 most frequently
`maxlen`	maximum blocksize read at once

Details

It is a simple http client. So it does not take care of special chars e.g. whitspaces. For details see e.g. perl HTTP::Request::Common documentation and `http://www.w3.org/Protocols/rfc1341/0_TableOfContents.html` The output is gathered in several blocks, if necessary. The parameter maxlen is used to determine the last data block by its length. The first block with less than maxlen bytes is considered the last one.

Value

The document which the server returns as a string.

Author(s)

E.W.Wolski wolski@molgen.mpg.de, Andreas Westfeld

See Also

getToHost, postToHost

Examples

```
host <- "api.scb.se"
path <- "/OV0104/v1/doris/en/ssd"
data <- '{"format":"json"}'
simplePostToHost(host, path, data, contenttype="text/json")
```

Index

getToHost, 2
getToHost2, 3

postToHost, 4

simplePostToHost, 5