Package 'cgir'

April 25, 2018

Title Support for using R as CGI scripts

Version 0.0.0.9000

Description What the package does (one paragraph).

Depends R (>= 3.2.3)

Imports RJSONIO

License GPL

Encoding UTF-8

LazyData true

Suggests testthat

RoxygenNote 6.0.1

R topics documented:

`interpolate` *Interpolate variables in a file*

Description

This function reads the contents of a file and performs a search-and-replace on a set of given variable names with given values. In the context of a CGI script, this function is useful for interpolating HTML templates and using them as the HTTP response. variable occurrences are marked with a sentinel character, which defaults to "%".

Usage

```
interpolate(filename, values, sentinel = "%")
```

Arguments

`filename`	the name of the file containing the text to be interpolated
`values`	a named vector, in which the names are used as the variable names, and the values as the variable values.
`sentinel`	a sentinel character indicating a variable in the text. Defaults to "%".

Details

For example, given the text "Hello %foo" and the variable foo="World", the interpolation will produce the text "Hello World"

Value

the interpolated text.

Examples

```
error <- "Insufficient amount of coffee!"
errorHTML <- interpolate("../../html/app/error.html",c(message=error))
respondHTML(errorHTML)
```

`makeUUID` *Create universally unique ID (UUIDv4)*

Description

Creates a universally unique identifier compatible with the UUID v4.0 standard. See `https://en.wikipedia.org/wiki/Universally_unique_identifier#Version_4_(random)`

Usage

```
makeUUID()
```

Value

the UUID as a character string

---

readGET *Read GET data from HTTP request*

Description

Reads the GET data transmitted via an HTTP request and returns it as a named character vector.

Usage

```
readGET()
```

Value

a named list containing the GET data

Examples

```
getData <- readGET()
if ("foo" %in% names(getData)) {
   foo <- getData[["foo"]]
}
```

---

readPOST *Read POST data from HTTP request*

Description

Reads the POST data transmitted via an HTTP request and returns it as a named character vector.

Usage

```
readPOST()
```

Value

a named list containing the POST data or NULL if none exists

Examples

```
postData <- readPOST()
if ("foobar" %in% names(postData)) {
   foobar <- postData[["foobar"]]
}
```

`respond` *Respond to HTTP request using a given MIME type*

Description

Responds to an HTTP request using a given MIME type

Usage

```
respond(content, mime)
```

Arguments

`content`	a character string containing the content
`mime`	a character string containing the MIME type (e.g. "text/html" or "image/gif")

Examples

```
error <- "Insufficient amount of coffee!"
errorHTML <- interpolate("../../html/app/error.html",c(message=error))
respondHTML(errorHTML)
```

`respond400` *Respond to HTTP request with a 400:BadRequest error*

Description

Responds to an HTTP request with 400:BadRequest error

Usage

```
respond400(message)
```

Arguments

`message`	the error message

`respond404` *Respond to HTTP request with a 404:NotFound error*

Description

Responds to an HTTP request with 404:NotFound error

Usage

```
respond404(message)
```

Arguments

`message`	the error message

`respondBinary` *Respond to HTTP request using binary data in a given MIME type*

Description

Responds to an HTTP request using binary data a given MIME type

Usage

```
respondBinary(binFile, mime)
```

Arguments

`binFile`	the name of a file containing the binary data
`mime`	a character string containing the MIME type (e.g. "image/gif")

Examples

```
error <- "Insufficient amount of coffee!"
errorHTML <- interpolate("../../html/app/error.html",c(message=error))
respondHTML(errorHTML)
```

`respondHTML` *Respond to HTTP request using HTML*

Description

Responds to an HTTP request using a given HTML content

Usage

```
respondHTML(html)
```

Arguments

`html`	a character string containing the HTML content

Examples

```
error <- "Insufficient amount of coffee!"
errorHTML <- interpolate("../../html/app/error.html",c(message=error))
respondHTML(errorHTML)
```

`respondJSON` *Respond to HTTP request using JSON*

Description

Responds to an HTTP request using JSON formatted data

Usage

```
respondJSON(content)
```

Arguments

`content`	a named list, which will be serialized to JSON using RJSONIO

Examples

```
data <- list(foo="bar",baz=1:5)
respondJSON(data)
```

`respondPDF` *Respond to HTTP request with PDF data*

Description

Responds to an HTTP request with PDF data from a given PDF file

Usage

```
respondPDF(pdfFile)
```

Arguments

`pdfFile`	the name of a file containing the binary data

`respondPNG` *Respond to HTTP request with PNG image data*

Description

Responds to an HTTP request with PNG image data from a given PNG file

Usage

```
respondPNG(pngFile)
```

Arguments

`pngFile`	the name of a file containing the binary data

`respondTemplateHTML` *Respond to HTTP request using HTML interpolated from a template file*

Description

Responds to an HTTP request using a given HTML file as well as interpolated content

Usage

```
respondTemplateHTML(templateFile, values)
```

Arguments

`templateFile`	the name of the file containing the HTML content
`values`	a named vector, in which the names are used as the variable names, and the values as the variable values.

Examples

```
error <- "Insufficient amount of coffee!"
respondTemplateHTML("../../html/app/error.html",c(message=error))
```

`respondTEXT` *Respond to HTTP request using plain text*

Description

Responds to an HTTP request using plain text

Usage

```
respondTEXT(text)
```

Arguments

`content`	a character string to respond

Examples

```
respondTEXT("Hello World!")
```

`setMessageSink` *Redirect messages to a log file*

Description

Redirects any messages (e.g. errors) to a log file. Doing so at the start of your script is absolutely vital! Otherwise any potential info messages, warnings or errors can corrupt your HTTP response

Usage

```
setMessageSink(filename = "msg.log")
```

Arguments

`filename`	the log file to which messages will be redirected. Must be writable for apache user!

Index