

# Package ‘r2shortcode’

May 9, 2026

**Type** Package

**Title** Shorten Function Names of Functions in Another Package and  
Create an Index to Make Them Accessible

**Version** 0.2

**Author** Obinna Obianom

**Maintainer** Obinna Obianom <idonshayo@gmail.com>

**Description** When creating a package, authors may sometimes struggle with coming up with easy and straightforward function names, and at the same time hoping that other packages do not already have the same function names. In trying to meet this goal, sometimes, function names are not descriptive enough and may confuse the potential users. The purpose of this package is to serve as a package function short form generator and also provide shorthand names for other functions. Having this package will entice authors to create long function names without the fear of users not wanting to use their packages because of the long names. In a way, everyone wins - the authors can use long descriptive function names, and the users can use this package to make short functions names while still using the package in question.

**License** MIT + file LICENSE

**URL** <https://github.com/oobianom/r2shortcode>

**BugReports** <https://github.com/oobianom/r2shortcode>

**Depends** R (> 3.6)

**Imports** utils, stringr

**Suggests** rmarkdown, knitr, qpdf

**Encoding** UTF-8

**VignetteBuilder** knitr

**Language** en-US

**LazyData** false

**RoxygenNote** 7.2.3

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2024-04-05 07:40:02 UTC

## Contents

chooseShortName . . . . .	2
clearStoredNames . . . . .	3
discardShortcodes . . . . .	3
hasSpecialCharacters . . . . .	4
hasUpperCase . . . . .	5
help . . . . .	5
index . . . . .	6
isUpperCase . . . . .	7
nameAlreadyExists . . . . .	7
searchNameSaveName . . . . .	8
shortenPkg . . . . .	9
storeChosenName . . . . .	9
whatis . . . . .	10

<b>Index</b>	<b>11</b>
--------------	-----------

---

chooseShortName	<i>Short algorithm to choose a short name based on a given long name</i>
-----------------	--

---

### Description

A very simple and short algorithm to choose a short name based on a given name

### Usage

```
chooseShortName(
  fullname = stop("Invalid string name entered."),
  withPrefix = NULL,
  withSuffix = NULL,
  envir = NULL,
  silent = FALSE
)
```

### Arguments

fullname	The name you intend to shorten
withPrefix	Prefix to include in front of the new short name
withSuffix	Suffix to include in front of the new short name
envir	The environment where to store the name
silent	Return response at the end of evaluations?

### Value

Short forms of functions

**Examples**

```
long_function_name <- 'longFunctionCall'  
short_function_name <- chooseShortName(long_function_name)  
short_function_name # the result should "lFC"
```

---

clearStoredNames	<i>Clears previously stored names</i>
------------------	---------------------------------------

---

**Description**

Beware that by clearing all stored names, you may inadvertently duplicate new shortnames

**Usage**

```
clearStoredNames(w = "all")
```

**Arguments**

w	what to clear
---	---------------

**Value**

empty stores for chosen name

**Examples**

```
nametostore = "ujuo"  
storeChosenName(nametostore) #store the chosen name  
nameAlreadyExists(nametostore) #check if the chosen name now exists in store  
clearStoredNames("all") #clear storage of all names  
nameAlreadyExists(nametostore) #check if the chosen name now exists in store, it should not
```

---

discardShortcodes	<i>Discard shortcodes</i>
-------------------	---------------------------

---

**Description**

This will discard all shorthand functions created and delete them from stores as well. Good if you inadvertently shorthand a package.

**Usage**

```
discardShortcodes(  
  pkg,  
  reflib = options()$.funcNamesPkgReference,  
  response = TRUE  
)
```

**Arguments**

pkg	package name
reflib	reference library, preferably leave unentered
response	TRUE or FALSE, return a response upon completion

**Value**

unloads short function names

**Examples**

```
pkgName = 'quickcode'  
shortenPkg(pkgName)  
discardShortcodes(pkgName)
```

---

hasSpecialCharacters *Does string have special characters?*

---

**Description**

Evaluates if a specified string contains special characters

**Usage**

```
hasSpecialCharacters(string)
```

**Arguments**

string	The string to evaluate
--------	------------------------

**Value**

TRUE or FALSE

**Examples**

```
strToTest1 <- 'obi_%_good_^you'  
strToTest2 <- 'obigoodyou'  
hasSpecialCharacters(strToTest1)  
hasSpecialCharacters(strToTest2)
```

---

hasUpperCase	<i>Contains uppercase?</i>
--------------	----------------------------

---

**Description**

Simply change if there is any uppercase letter in a string

**Usage**

```
hasUpperCase(string)
```

**Arguments**

string            the string to evaluate

**Value**

TRUE or FALSE if the string has an upper case letter

**Examples**

```
strToTest1 <- 'obiWentToSchool'  
strToTest2 <- 'obiwenttoschool'  
hasUpperCase(strToTest1)  
hasUpperCase(strToTest2)
```

---

help	<i>Help for all functions</i>
------	-------------------------------

---

**Description**

Access help for all functions including any newly created shorthand functions

**Usage**

```
help(functionName, package = NULL)
```

**Arguments**

functionName    function name to search for  
 package        package name containing the function. Leave unentered if this is a shorthand function

**Value**

Documentation for function

**Examples**

```
pkgname <- 'qpdf' #package name
shortenPkg(pkgname,TRUE) #shorten the package
index(pkgname) #index the package functions shortened
help('qpd.pl') #choose a function name and find help
```

---

 index

---

*Index a shortened package*


---

**Description**

This function will provide you the list of shorthand functions created for a package

**Usage**

```
index(pkg = stop("Enter a package name to index"))
```

**Arguments**

pkg            The package name

**Value**

List of long and short forms of particular functions

**Examples**

```
if(interactive()){
  pkgname <- 'qpdf'
  shortenPkg(pkgname,TRUE)
  index(pkgname)
}
```

---

isUpperCase	<i>Is string uppercase?</i>
-------------	-----------------------------

---

**Description**

Simply, test if a string is uppercase and return TRUE or FALSE

**Usage**

```
isUpperCase(string)
```

**Arguments**

string	The string to evaluate
--------	------------------------

**Value**

TRUE or FALSE if the string is all upper case

**Examples**

```
strTest1 <- 'OBI'  
strTest2 <- 'obi0'  
isUpperCase(strTest1)  
isUpperCase(strTest2)
```

---

nameAlreadyExists	<i>Does name already exist in memory?</i>
-------------------	---

---

**Description**

Evaluates if a name has already been saved by the r2shortcode. Keep in mind that if you previously used the clear function, previously saved names will be cleared

**Usage**

```
nameAlreadyExists(name)
```

**Arguments**

name	The name to lookup
------	--------------------

**Value**

TRUE or FALSE

**Examples**

```
nameToCheck <- 'Obinna'  
nameAlreadyExists(nameToCheck)
```

---

searchNameSaveName      *Search for the existence of a name and save if unavailable*

---

**Description**

Carries out a search on the already used shorthand function names and if the new name does not exist, it saves it

**Usage**

```
searchNameSaveName(withPrefix = NULL, strN.complete, withSuffix = NULL)
```

**Arguments**

withPrefix      Provide the prefix to use for the function name, could use NULL  
strN.complete    Provide the function name to search for, and save if feasible  
withSuffix      Provide the suffix to use for the function name, could use NULL

**Value**

Concatenate of search and a boolean

**Examples**

```
nameToCheck <- 'ObiObianom'  
searchNameSaveName(strN.complete= nameToCheck)  
searchNameSaveName(strN.complete= nameToCheck)
```

---

shortenPkg	<i>Shorten the package</i>
------------	----------------------------

---

**Description**

This will create shorthand functions for functions in a package. This function brings together most functions in this package.

**Usage**

```
shortenPkg(pkg, addPrefix = TRUE, silent = FALSE, num.prefix = 3)
```

**Arguments**

pkg	package name
addPrefix	should prefix be added to the new names, TRUE or FALSE. Prefix will by default be first 3 letters of the package name
silent	return messages at the end of evaluation, TRUE or FALSE
num.prefix	if you specify to addPrefix, how many letters will you like to add?

**Value**

short function names for all the functions in the package

**Examples**

```
pkg <- 'qpdf'  
shortenPkg(pkg, FALSE, TRUE)
```

---

storeChosenName	<i>Store chosen name in memory</i>
-----------------	------------------------------------

---

**Description**

When all else passes, store this particular name in memory

**Usage**

```
storeChosenName(name)
```

**Arguments**

name	The chosen name to store
------	--------------------------

**Value**

store the chosen name in storage variable

**Examples**

```
nameStore <- 'ObiStore1'  
storeChosenName(nameStore)
```

---

whatis

*Help for all functions*

---

**Description**

Access help for all functions including any newly created shorthand functions

**Usage**

```
whatis(functionName, package = NULL)
```

**Arguments**

functionName	function name to search for
package	package name containing the function. Leave unentered if this is a shorthand function

**Value**

help for the particular function

**Examples**

```
pkgname <- 'qpdf' #package name  
shortenPkg(pkgname,TRUE) #shorten the package  
index(pkgname) #index the package functions shortened  
whatis('qpd.pl') #choose a function name and find help
```

# Index

chooseShortName, [2](#)  
clearStoredNames, [3](#)  
  
discardShortcodes, [3](#)  
  
hasSpecialCharacters, [4](#)  
hasUpperCase, [5](#)  
help, [5](#)  
  
index, [6](#)  
isUpperCase, [7](#)  
  
nameAlreadyExists, [7](#)  
  
searchNameSaveName, [8](#)  
shortenPkg, [9](#)  
storeChosenName, [9](#)  
  
whatis, [10](#)