

# Package ‘whisker’

May 8, 2026

**Maintainer** Edwin de Jonge <edwindjonge@gmail.com>

**License** GPL-3

**Title** {{mustache}} for R, Logicless Templating

**Type** Package

**LazyLoad** yes

**Author** Edwin de Jonge

**Description** Implements 'Mustache' logicless templating.

**Version** 0.4.1

**URL** <https://github.com/edwindj/whisker>

**Suggests** markdown

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2022-12-05 15:33:50 UTC

## Contents

whisker-package . . . . .	2
iteratelist . . . . .	3
rowSplit . . . . .	4
whisker.escape . . . . .	5
whisker.render . . . . .	5
<b>Index</b>	<b>7</b>

## Description

Whisker is a templating engine for R conforming to the Mustache specification. Mustache is a logicless templating language, meaning that no programming source code can be used in your templates. This may seem very limited, but Mustache is nonetheless powerful and has the advantage of being able to be used unaltered in many programming languages. For example it make it very easy to write a web application in R using Mustache templates and where the browser can template using javascript's "Mustache.js"

## Details

Mustache (and therefore whisker) takes a simple but different approach to templating compared to most templating engines. Most templating libraries for example Sweave and brew allow the user to mix programming code and text throughout the template. This is powerful, but ties a template directly to a programming language. Furthermore that approach makes it difficult to separate programming code from templating code.

Whisker on the other hand, takes a Mustache template and uses the variables of the current environment (or the supplied list) to fill in the variables.

## Examples

```
template <-
'Hello {{name}}
You have just won ${{value}}!
{{#in_ca}}
Well, ${{taxed_value}}, after taxes.
{{/in_ca}}'

data <- list( name = "Chris"
              , value = 10000
              , taxed_value = 10000 - (10000 * 0.4)
              , in_ca = TRUE
              )

whisker.render(template, data)
```

```
base <-
'<h2>Names</h2>
{{#names}}
  {> user}
{{/names}}'

user <- '<strong>{{name}}</strong>'
```

```
names <- list(list(name="Alice"), list(name="Bob"))  
whisker.render(base, partials=list(user=user))
```

---

**iteratelist***Create an iteration list from a R object*

---

### Description

In some case it is useful to iterate over a named list or vector `iteratelist` will create a new unnamed list with name value members: each item will be a list where 'name' is the corresponding name and 'value' is the original value in list x.

### Usage

```
iteratelist(x, name = "name", value = "value")
```

### Arguments

x	list or other object that will be coerced to list
name	character name for resulting name member.
value	character name for resulting value member.

### Value

unnamed list with name value lists

### Examples

```
# create an iteration list from a named vector  
x <- c(a=1, b=2)  
iteratelist(x)  
  
# iterate over the members of a list  
x <- list(name="John", age="30", gender="male")  
iteratelist(x, name="variable")  
  
# iterate over an unnamed vector  
values <- c(1,2,3,4)  
  
template <-  
'{{#values}}  
* Value: {{.}}  
{{/values}}'  
  
whisker.render(template)
```

```
#or

values <- iteratelist(values, value="count")

template <-
'{{#values}}
* Value: {{count}}
{{/values}}'

whisker.render(template)
```

---

**rowSplit***Split a data.frame or matrix into rows*

---

### Description

Utility function for splitting a data.frame into rows. In a whisker template it can be useful to iterate over the rows of a data.frame or matrix. For example rendering a table in HTML.

### Usage

```
rowSplit(x, ...)
```

### Arguments

x	data.frame or matrix
...	other options will be passed onto <a href="#">split</a>

### Examples

```
dat <- head(InsectSprays)
dat <- unname(rowSplit(dat))

template <-
"{{#dat}}
count: {{count}}, spray: {{spray}}\n
{{/dat}}"

whisker.render(template)
```

---

whisker.escape	<i>escape basic HTML characters</i>
----------------	-------------------------------------

---

**Description**

This method is called for normal mustache keys

**Usage**

```
whisker.escape(x)
```

**Arguments**

x	character that will be escaped
---	--------------------------------

**Value**

HTML escaped character

---

whisker.render	<i>Logicless templating</i>
----------------	-----------------------------

---

**Description**

Logicless templating

**Usage**

```
whisker.render(template, data = parent.frame(), partials = list(),  
  debug = FALSE, strict = TRUE)
```

**Arguments**

template	character with template text
data	named list or environment with variables that will be used during rendering
partials	named list with partial templates, will be used during template construction
debug	Used for debugging purposes, likely to disappear
strict	logical if TRUE the seperation symbol is a "." otherwise a "\$"

**Value**

character with rendered template

**Note**

By default whisker applies html escaping on the generated text. To prevent this use `{{{variable}}}` (triple) in stead of `{{variable}}`.

**Examples**

```
template <- "Hello {{place}}!"
place <- "World"

whisker.render(template)

# to prevent html escaping use triple {{{}}}
template <-
  "I'm escaped: {{name}}
  And I'm not: {{{name}}}"

data <- list( name = '<My Name="Nescio">')
whisker.render(template, data)
# I'm escaped: &lt;My Name="Nescio&amp;"&gt;
# And I'm not: <My Name="Nescio">
```

# Index

`iteratelist`, 3

`rowSplit`, 4

`split`, 4

`whisker-package`, 2

`whisker.escape`, 5

`whisker.render`, 5