

Package ‘sticky’

July 23, 2025

Type Package

Title Persist Attributes Across Data Operations

Description In base R, object attributes are lost when objects are modified by common data operations such as subset, filter, slice, append, extract etc. This packages allows objects to be marked as 'sticky' and have attributes persisted during these operations or when inserted into or extracted from list-like or table-like objects.

Version 0.5.6.1

Date 2020-03-15

License GPL-2 | file LICENSE

URL <https://github.com/decisionpatterns/sticky>
<http://decisionpatterns.com>

BugReports <https://github.com/decisionpatterns/sticky/issues>

Depends R (>= 3.1.0)

Imports methods

Suggests magrittr (>= 1.5), testthat (>= 0.10.0), data.table (>= 1.9.6), tibble (>= 1.3.3), knitr, rmarkdown

LazyData TRUE

Collate 'init.R' 'sticky.R' 'append.R' 'extract1.R' 'print.R'
'subset.R' 'utils.R' 'zzz.R'

VignetteBuilder knitr

RoxygenNote 7.0.2

Repository CRAN

Encoding UTF-8

Language en-US

NeedsCompilation no

Author Christopher Brown [aut, cre],
Decision Patterns [cph]

Maintainer Christopher Brown <chris.brown@decisionpatterns.com>

Date/Publication 2020-03-16 17:40:02 UTC

Contents

.init	2
append	3
print.sticky	3
sticky	4
subset.sticky	5
[.sticky	6
Index	7

.init	<i>Internal, non-exported unit functions</i>
-------	--

Description

These are internal functions for the sticky package. They are not exported and should not be used directly. Instead use the proper interface, [sticky](#) and [unstick](#)

Usage

```
.init(x)

.deinit(x)
```

Arguments

x vector

Details

.init : initializes the class .deinit : uninitialize the class
These are internals functions not to be called directly, please use the interfaces, [sticky](#) and [unsticky](#).

See Also

[sticky](#) and [unsticky](#)

Examples

```
# - tk
```

append	<i>Append elements to sticky object</i>
--------	---

Description

Append elements to sticky object

Usage

append(x, ...)

Arguments

- x sticky object
- ... additional arguments to base::append
append preserves object attributes by wrapping base::append and copying objects to the result.

See Also

[append](#)

print.sticky	<i>print a sticky attribute object</i>
--------------	--

Description

print a sticky object which mostly delegates it the the next method.

Usage

```
## S3 method for class 'sticky'  
print(x, ...)
```

Arguments

- x sticky object
- ... further arguments passed to or from other methods.

Details

prints a sticky object; this is also the default method for sticky objects

Value

An invisible copy of x.

Examples

```
x <- sticky(1:3)
print(x)
x
```

sticky	<i>sticky</i>
--------	---------------

Description

Persist Attributes

Usage

sticky(x)

unstick(x)

unsticky(x)

is.sticky(x)

sticky_all(x)

Arguments

x objects to have attributes preserved.

Details

sticky creates objects whose attributes are resilient to subsetting and placing in containers such as lists, data.frames and data.tables.

References

This has been asked and written about numerous times on Stack Overflow. The first has a fairly comprehensive list to others SO discussions.

<http://stackoverflow.com/questions/23991060/loss-of-attributes-despite-attempts-to-preserve-them>
<http://stackoverflow.com/questions/10404224/how-to-delete-a-row-from-a-data-frame-without-losing-t>
<http://stackoverflow.com/questions/23991060/loss-of-attributes-despite-attempts-to-preserve-them>
<http://stackoverflow.com/questions/23841387/approaches-to-preserving-objects-attributes-during-ext>
<http://stackoverflow.com/questions/13432519/indexing-operation-removes-attributes>

Examples

```

x <- 1:5
attr(x, 'foo') <- 'bar'
x <- x[1:3]
attr(x, 'foo')          # NULL

x <- sticky(1:5)
attr(x, 'foo') <- 'bar'
x_1 <- x[1:3]
attr(x_1, 'foo')        # bar

x_2 <- unstick(x)
x_2 <- x[1:3]
attr(x_2, 'foo')        # NULL

is.sticky(x)            # TRUE

```

subset.sticky	<i>Subset elements to sticky object</i>
---------------	---

Description

Subset elements to sticky object

Usage

```
## S3 method for class 'sticky'
subset(x, ...)
```

Arguments

x	sticky object
...	additional arguments to <code>base::subset</code> subset preserves object attributes by wrapping <code>base::subset</code> and copying objects to the result.

See Also

[subset](#)

`[.sticky]`*Extract*

Description

Extract

Usage

```
## S3 method for class 'sticky'  
x[...]
```

Arguments

<code>x</code>	sticky; object to extract from
<code>...</code>	additional parameters

Details

Implements '[' for sticky attributes. This methods ensures that existing attributes are propagated forward.

See Also

[attributes](#), specifically `mostattributes`

Index

`.deinit(.init)`, 2
`.init`, 2
`[.sticky]`, 6

`append`, 3, 3
`attributes`, 6

`is.sticky(sticky)`, 4

`print.sticky`, 3

`sticky`, 2, 4
`sticky_all(sticky)`, 4
`subset`, 5
`subset(subset.sticky)`, 5
`subset.sticky`, 5

`unstick`, 2
`unstick(sticky)`, 4
`unsticky(sticky)`, 4