Package 'postlogic'

May 9, 2026

Type Package

Title Infix and Postfix Logic Operators

Version 0.1.0.1

Maintainer Andrew Redd <Andrew.Redd@hsc.utah.edu>

Description Provides adds postfix and infix logic operators for if, then, unless, and otherwise.

Language en-US

License GPL-2

Encoding UTF-8

LazyData true

Suggests testthat, covr

URL https://github.com/RDocTaskForce/postlogic

BugReports https://github.com/RDocTaskForce/postlogic/issues

NeedsCompilation no

Author Andrew Redd [aut, cre] (ORCID: <https://orcid.org/0000-0002-6149-2438>)

Repository CRAN

Date/Publication 2019-12-18 09:15:26 UTC

Contents

if-otherwise *Postfix if-otherwise logic*

Description

This construction allows logical statements to be placed after the value to be returned. Take note that the ‘ as other custom infix operators and so care should be taken that the effect is as desired.

Usage

```
prior %if% proposition

prior %if% proposition %otherwise% alternate
```

Arguments

`prior`	The value to be returned if proposition evaluates to TRUE.
`proposition`	The logical statement to evaluate
`alternate`	The value to be returned if proposition evaluates to FALSE.
`prior %if% proposition`	An `%if%` statement.

See Also

Other postlogic: `unless-then`

Examples

```
x <- 1
x <- (x+1) %if% is.numeric(x) %otherwise% "Hmm this isn't right O.o"
x # 2

x <- 1i
x <- (x+1) %if% is.numeric(x) %otherwise% "Hmm this isn't right O.o"
x # Hmm this isn't right
```

unless-then *Infix unless-then logic*

Description

These give logic that can be used as a qualifying statement that occurs after the value statement. Take note that the ‘ as other custom infix operators and so care should be taken that the effect is as desired.

Usage

```
prior %unless% proposition
prior %unless% proposition %then% alternate
```

Arguments

`prior`	Value to be returned unless proposition returns FALSE.
`proposition`	The logical statement to condition on.
`alternate`	When proposition returns true and the the alternate value is returned.
`prior %unless% proposition`	An `%if%` statement.

See Also

Other postlogic: `if-otherwise`

Examples

```
x <- 4
x <- sqrt(x) %unless% is.complex(x) %then% "This is too hard :("
x # 2

x <- 4i
x <- sqrt(x) %unless% is.complex(x) %then% "This is too hard :("
x # This is too hard :(
```

Index