

Logical Operators

Description

These operators act on logical vectors.

Usage

```
! x
x & y
x && y
x | y
x || y
xor(x, y)
```

```
isTRUE(x)
```

Arguments

`x`, `y` logical vectors, or objects which can be coerced to such or for which methods have been written.

Details

`!` indicates logical negation (NOT).

`&` and `&&` indicate logical AND and `|` and `||` indicate logical OR. The shorter form performs elementwise comparisons in much the same way as arithmetic operators. The longer form evaluates left to right examining only the first element of each vector. Evaluation proceeds only until the result is determined. The longer form is appropriate for programming control-flow and typically preferred in [if](#) clauses.

`xor` indicates elementwise exclusive OR.

`isTRUE(x)` is an abbreviation of `identical(TRUE, x)`, and so is true if and only if `x` is a length-one logical vector with no attributes (not even names).

Numeric and complex vectors will be coerced to logical values, with zero being false and all non-zero values being true. Raw vectors are handled without any coercion for `!`, `&` and `|`, with these operators being applied bitwise (so `!` is the 1-complement).

The operators `!`, `&` and `|` are generic functions: methods can be written for them individually or via the [Ops](#) group generic function. (See [Ops](#) for how dispatch is computed.)

[NA](#) is a valid logical object. Where a component of `x` or `y` is `NA`, the result will be `NA` if the outcome is ambiguous. In other words `NA & TRUE` evaluates to `NA`, but `NA & FALSE` evaluates to `FALSE`. See the examples below.

See [Syntax](#) for the precedence of these operators: unlike many other languages (including S) the AND and OR operators do not have the same precedence (the AND operators are higher than the OR operators).

Value

For `!`, a logical or raw vector of the same length as `x`.

For `|`, `&` and `xor` a logical or raw vector. The elements of shorter vectors are recycled as necessary (with a [warning](#) when they are recycled only *fractionally*). The rules for determining the attributes of the result are rather complicated. Most attributes are taken from the longer argument, the first if they are of the same length. Names will be copied from the first if it is the same length as the answer, otherwise from the second if that is. For time series, these operations are allowed only if the series are compatible, when the class and `tsp` attribute of whichever is a time series (the same, if both are) are used. For arrays (and an array result) the dimensions and dimnames are taken from first argument if it is an array, otherwise the second.

For `||`, `&&` and `isTRUE`, a length-one logical vector.

S4 methods

`!`, `&` and `|` are S4 generics, the latter two part of the `Logic` group generic (and hence methods need argument names `e1`, `e2`).

Prior to R 2.6.0 S4 methods for `!` needed argument name `e1`, but now `x` is correct.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

[TRUE](#) or [logical](#).

[any](#) and [all](#) for OR and AND on many scalar arguments.

[Syntax](#) for operator precedence.

Examples

```
y <- 1 + (x <- stats::rpois(50, lambda=1.5) / 4 - 1)
x[(x > 0) & (x < 1)]      # all x values between 0 and 1
if (any(x == 0) || any(y == 0)) "zero encountered"

## construct truth tables :

x <- c(NA, FALSE, TRUE)
names(x) <- as.character(x)
outer(x, x, "&")## AND table
outer(x, x, "|")## OR table
```

[Package *base* version 2.7.1 [Index](#)]