|  |  |
| --- | --- |
| numeric {base} | R Documentation |

**Numeric Vectors**

**Description**

Creates or coerces objects of type "numeric". is.numeric is a more general test of an object being interpretable as numbers.

**Usage**

numeric(length = 0)

as.numeric(x, ...)

is.numeric(x)

**Arguments**

|  |  |
| --- | --- |
| length | A non-negative integer specifying the desired length. Double values will be coerced to integer: supplying an argument of length other than one is an error. |
| x | object to be coerced or tested. |
| ... | further arguments passed to or from other methods. |

**Details**

numeric is identical to [double](http://127.0.0.1:14695/library/base/help/double) (and real). It creates a double-precision vector of the specified length with each element equal to 0.

as.numeric is a generic function, but S3 methods must be written for [as.double](http://127.0.0.1:14695/library/base/help/as.double). It is identical to as.double.

is.numeric is an [internal generic](http://127.0.0.1:14695/library/base/help/internal%20generic) primitive function: you can write methods to handle specific classes of objects, see [InternalMethods](http://127.0.0.1:14695/library/base/help/InternalMethods). It is **not** the same as [is.double](http://127.0.0.1:14695/library/base/help/is.double). Factors are handled by the default method, and there are methods for classes "[Date](http://127.0.0.1:14695/library/base/help/Date)", "[POSIXt](http://127.0.0.1:14695/library/base/help/POSIXt)" and "[difftime](http://127.0.0.1:14695/library/base/help/difftime)" (all of which return false). Methods for is.numeric should only return true if the base type of the class is double or integer *and* values can reasonably be regarded as numeric (e.g., arithmetic on them makes sense, and comparison should be done via the base type).

**Value**

for numeric and as.numeric see [double](http://127.0.0.1:14695/library/base/help/double).

The default method for is.numeric returns TRUE if its argument is of [mode](http://127.0.0.1:14695/library/base/help/mode) "numeric" ([type](http://127.0.0.1:14695/library/base/help/type) "double" or type "integer") and not a factor, and FALSE otherwise. That is, is.integer(x) || is.double(x), or (mode(x) == "numeric") && !is.factor(x).

**Warning**

If x is a [factor](http://127.0.0.1:14695/library/base/help/factor), as.numeric will return the underlying numeric (integer) representation, which is often meaningless as it may not correspond to the factor [levels](http://127.0.0.1:14695/library/base/help/levels), see the ‘Warning’ section in [factor](http://127.0.0.1:14695/library/base/help/factor) (and the 2nd example below).

**S4 methods**

as.numeric and is.numeric are internally S4 generic and so methods can be set for them *via* setMethod.

To ensure that as.numeric and as.double remain identical, S4 methods can only be set for as.numeric.

**Note on names**

It is a historical anomaly that **R** has two names for its floating-point vectors, [double](http://127.0.0.1:14695/library/base/help/double) and [numeric](http://127.0.0.1:14695/library/base/help/numeric) (and formerly had real).

double is the name of the [type](http://127.0.0.1:14695/library/base/help/type). numeric is the name of the [mode](http://127.0.0.1:14695/library/base/help/mode) and also of the implicit [class](http://127.0.0.1:14695/library/base/help/class). As an S4 formal class, use "numeric".

The potential confusion is that **R** has used [*mode*](http://127.0.0.1:14695/library/base/help/mode) "numeric" to mean ‘double or integer’, which conflicts with the S4 usage. Thus is.numeric tests the mode, not the class, but as.numeric (which is identical to as.double) coerces to the class.

**References**

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

**See Also**

[double](http://127.0.0.1:14695/library/base/help/double), [integer](http://127.0.0.1:14695/library/base/help/integer), [storage.mode](http://127.0.0.1:14695/library/base/help/storage.mode).

**Examples**

as.numeric(c("-.1"," 2.7 ","B")) # (-0.1, 2.7, NA) + warning

as.numeric(factor(5:10)) # not what you'd expect

f <- factor(1:5)

## what you typically meant and want:

as.numeric(as.character(f))

## the same, considerably (for long factors) more efficient:

as.numeric(levels(f))[f]