

Package: rtype (via r-universe)

September 12, 2024

Type Package

Title A strong type system for R

Version 0.1-1

Author Kun Ren <ken@renkun.me>

Maintainer Kun Ren <ken@renkun.me>

Description A strong type system for R which supports symbol declaration and assignment with type checking and condition checking.

Depends R (>= 2.15)

Date 2014-08-15

Suggests testthat, knitr

License MIT + file LICENSE

URL <http://renkun.me/rtype>, <https://github.com/renkun-ken/rtype>

BugReports <https://github.com/renkun-ken/rtype/issues>

Roxygen list(wrap = FALSE)

ByteCompile TRUE

Repository <https://renkun-ken.r-universe.dev>

RemoteUrl <https://github.com/renkun-ken/rtype>

RemoteRef HEAD

RemoteSha 1bdfb129002c89619b02ed120202ad1a2b7c48da

Contents

declare	2
typed-assign	2

Index	5
--------------	----------

declare *Declare symbols*

Description

Declare symbols

Usage

```
declare(..., .envir = parent.frame())
```

Arguments

... Symbols to declare
.envir environment to store the symbols

Examples

```
declare(x,y=numeric(),z=integer())
```

typed-assign *Assign with type checking*

Description

Assign with type checking

Usage

```
atomic(x, ...) <- value  
integer(x, ...) <- value  
numeric(x, ...) <- value  
double(x, ...) <- value  
logical(x, ...) <- value  
character(x, ...) <- value  
raw(x, ...) <- value  
complex(x, ...) <- value
```

```
matrix(x, ...) <- value
array(x, ...) <- value
list(x, ...) <- value
pairlist(x, ...) <- value
envir(x, ...) <- value
name(x, ...) <- value
symbol(x, ...) <- value
call(x, ...) <- value
factor(x, ...) <- value
fun(x, ...) <- value
expression(x, ...) <- value
language(x, ...) <- value
object(x, ...) <- value
table(x, ...) <- value
recursive(x, ...) <- value
vector(x, ...) <- value
data.frame(x, ...) <- value
null(x, ...) <- value
check(x, ...) <- value
```

Arguments

x	symbol
...	additional conditions taking the following forms: <ol style="list-style-type: none">1. fun = v, i.e. fun(x) must be equal v.2. cond, i.e. cond(x) must be TRUE.3. a function like function(x) mean(x) <= 5.0
value	value to be assigned

Examples

```
## Not run:  
x <- 10L  
atomic(x) <- 20  
numeric(x) <- 10  
numeric(x, length = 10L) <- 1:10  
  
cond1 <- function(x) mean(x) <= 5  
numeric(x, cond1) <- 0:9  
  
## End(Not run)
```

Index

`array<-` (typed-assign), 2
`atomic<-` (typed-assign), 2

`call<-` (typed-assign), 2
`character<-` (typed-assign), 2
`check<-` (typed-assign), 2
`complex<-` (typed-assign), 2

`data.frame<-` (typed-assign), 2
`declare`, 2
`double<-` (typed-assign), 2

`envir<-` (typed-assign), 2
`expression<-` (typed-assign), 2

`factor<-` (typed-assign), 2
`fun<-` (typed-assign), 2

`integer<-` (typed-assign), 2

`language<-` (typed-assign), 2
`list<-` (typed-assign), 2
`logical<-` (typed-assign), 2

`matrix<-` (typed-assign), 2

`name<-` (typed-assign), 2
`null<-` (typed-assign), 2
`numeric<-` (typed-assign), 2

`object<-` (typed-assign), 2

`pairlist<-` (typed-assign), 2

`raw<-` (typed-assign), 2
`recursive<-` (typed-assign), 2

`symbol<-` (typed-assign), 2

`table<-` (typed-assign), 2
`typed-assign`, 2

`vector<-` (typed-assign), 2