Package 'typehint'

August 10, 2021

Title Auto-Check Types, Dimensions, and Values of Function Arguments

Description
Type hints are special comments within a function body indicating the intended nature of the function's arguments in terms of data types, dimensions and permitted values. The actual parameters with which the function is called are evaluated against these type hint comments at run-time.

Version 0.1.2

Maintainer Joachim Zuckarelli <joachim@zuckarelli.de>

License GPL-3

Encoding UTF-8

Imports stringr, crayon, rlist

Repository CRAN

BugReports https://github.com/jsugarelli/typehint/issues

URL https://github.com/jsugarelli/typehint/

RoxygenNote 7.1.1

NeedsCompilation no

Author Joachim Zuckarelli [aut, cre] (<https://orcid.org/0000-0002-9280-3016>)

Date/Publication 2021-08-10 09:10:02 UTC

R topics documented:

check_types *Type hints - Automatic checks of function arguments*

Description

The check_types() function is used within the body of a function to evaluate the parameters of a call to that function against the requirements defined in the type hint comments. See *Details* section for more information on type hint comments.

Usage

```
check_types(show.msg = TRUE, abort = TRUE, messages = NULL, color = "#bd0245")
```

Arguments

show.msg	Indicates if a message is shown whenever a check fails (default is TRUE).
abort	Indicates if checks are stopped after the first error occurred (default is TRUE), or if all checks are performed.
messages	A vector with five message templates to be used as error messages. NULL, if the default templates shall be used. Templates can make use of predefined placeholders to convey information important for understanding the source of the problem. See below for a comprehensive discussion of error messages.
color	Standard hex RGB color code of the error messages (default is "#bd0245").

Value

TRUE, if *all* parameter values provided in the function call pass *all* tests / adhere to all restrictions defined in the type hint commments, FALSE otherwise.

How do type hints work? Overview.

Type hints are special comments with a leading #| within a function body indicating the intended nature of the function's arguments in terms of data types, dimensions and even permitted values. The actual parameters with which the function is called can be evaluated against these type hint comments using the check_types() function.

check_types() returns FALSE if any of the checks fails. Checking can be aborted after the first error occurs, or it can be continued until all checks have been performed. Optionally, the user is shown a message indicating the nature of the problem with the function arguments. The messages can be completely customized using placerholder variables for all kinds of relevant information.

Type hint comments

Placement of comments:

Type hint comments always need to be placed *inside* a function body and refer to the arguments of that function. They can be placed *anywhere* in the function body (even after the call of check_types()). Type hint comments are regular R comments but start with #| (hash plus

pipe, without blanks in between). Each function argument will have its own type hint comment line. Type hint comments can cover a subset of all arguments, so there can be arguments without any type hint restrictions.

Comment syntax:

Type hint comments consist of a data type check and (optionally) dimension and value checks:

- **Data type check**: The data type checks for the data type of the argument. At this point, the data type check needs to be the first check in a type hint comment and can only comprise one permitted data type. The syntax is `argument_name data_type`. A valid type hint comment consisting only of a data type check could thus look like this: `#| degrees_celsius numeric`.
- **Dimension check**: The dimension check checks for the number and size of the dimensions of the argument. It is constructed using the `dim()` function. `dim()` takes one parameter per dimension of the argument. The parameters specify the size of each of the dimensions of the argument either as specific values or as comparisons. So, the general syntax is: `dim([comparison_operator]dimsize[,[comparison_operator]#' dim_size]*)`. For example, if the argument (called `unemployment`) is required to be a dataframe with exactly four columns and at least two rows then the type hint comment would look like this: `#| unemployment data.frame dim(>=2,4)`. When `check_types()` evaluates the parameters supplied in the function call it looks for the number of dimensions of the parameter as well as the size of each dimension.
- **Value check**: The value check evaluates the actual value of the parameter supplied in the function call and rejects the value if it is on an exclude list. Such exclude lists are constructed using the `not()` function. The `not()` function expects as its arguments the values that shall not be permitted as parameter values. These values can include `NA` and `NULL`. The general syntax of the `not()` function is: `not(excludevalue[,excludevalue]*)`. If we had an argument called `surname` and this argument must not be `NA` or`""` (empty character) then the required type hint check would like this: `#| surname character not("",NA)`. There can be multiple `not`-lists in each type hint comment. If the parameter supplied in the function call consists of, by its nature, multiple elements, like it is the case with dataframes, list, and matrices, then the value check fails if *any* element of the parameter provided in the function call is on the exclude list.

When formulating `dim` or `not` restrictions you can use the values of other parameters of the function call. So, if you have a function with two arguments, a number of children (`num.children`) and a numeric vector with the ages of these children (`age.children`) you can have a type hint comment for the latter which looks like this: `#| age.children numeric dim(num.children)`.

If a type hint check fails

If any of the checks fails `check_types()` returns `FALSE`, otherwise it returns `TRUE`. If `show.msg=TRUE` then a message will be shown in the R console. The messages can be customized using templates (see next section). Depending on the value of `abort` the checking of parameters is continued (`abort=FALSE`) or stopped immediately (`abort=TRUE`), i.e. no further checks are performed after the first error.

Type hint output messages

Message templates:

The error messages that are shown (if show.msg=TRUE) when a check fails are based on templates. The templates are provided to the check_types() function via the messages argument. messages is a character vector with five elements, one for each possible kind of error message (or NULL, if the default error messages shall be used); the types of error messages are:

- General intro message (default: "Problem in function '#fun()'")
- Wrong parameter type (default: "Argument '#arg' (#argval) is of class #type_is but needs to be of class #type_req.")
- Wrong dimension size of parameter (default: "Size of dimension #dimno of argument '#arg' must be #dimcomp#dim_req,but is actually #dim_is.")
- Wrong number of dimensions of parameter (default: "Number of dimensions of argument '#arg' must be #dimcnt_req but is actually #dimcnt_is.")
- Parameter value is not permitted (default: "#argval is not a valid value for argument #arg.")

The messages provided via the messages argument are templates that can use predefined placeholders to convey information relevant for understanding the problem.

Placeholder that can be used in message templates:

- *#fun*: The name of the function of which the parameter values are to be checked (i.e. the function check_types() is applied to)
- *#arg*: The name of the argument
- *#argval*: The value of the parameter used in the function call
- *#type_req*: The required type for the argument
- *#type_is*: The actual type of the parameter used in the function call
- *#dimcnt_req*: The required number of dimensions of the argument
- *#dimcnt_is*: The actual number of dimensions of the parameter used in the function call
- *#dim_req*: The required size of the dimension where a dimension size error occurred
- *#dim_is*: The actual size of the dimension where a dimension size error occurred
- *#dimcomp*: The comparison operator used in combination with #dim_req, the required size of the dimension (e.g. the >= in >=2, if this dimension of the argument is to be greater than 1)
- *#dimno*: The index of the dimension where a dimension size error occurred

See Also

Other typehint: show_typehints(), typehint

Examples

```
celsius_to_fahrenheit <- function(degrees_celsius) {
 #| degrees_celsius numeric dim(1) not(NA, NULL)

 if(check_types()) return(degrees_celsius * 9/5 + 32)
 else return(NA)

}
```

show_typehints *Type hints - Automatic checks of function arguments*

Description

Prints out the type hint restrictions for a function in the R console, based on the type hint comments within the function' body.

Usage

```
show_typehints(fun, color = "#bd0245")
```

Arguments

fun	The function of which the type hint checks will be shown.
color	Color of the output in standard hex RGB format, default is #bd0245.

Value

No return value, only output in the R console.

See Also

Other typehint: check_types(), typehint

Examples

```
celsius_to_fahrenheit <- function(degrees_celsius) {
 #| degrees_celsius numeric dim(1) not(NA, NULL)

 if(check_types()) return(degrees_celsius * 9/5 + 32)
 else return(NA)

}

show_typehints(celsius_to_fahrenheit)
```

typehint *Package 'typehint'*

Description

Automatically check the data type, number and size of dimensions, and values of function arguments with simple type hint comments in the function code.

Details

Type hints are special comments with a leading `#|` within a function body indicating the intended nature of the function's arguments in terms of data types, dimensions and even permitted values. The actual parameters with which the function is called can be evaluated against these type hint comments using the `check_types()` function.

Value

None.

Author

Joachim Zuckarelli, joachim@zuckarelli.de, @jsugarelli

See Also

Other typehint: `check_types()`, `show_typehints()`

Index