

Package ‘startupmsg’

November 3, 2009

Encoding latin1

Version 0.7

Date 2009-10-16

Title Utilities for start-up messages

Description Utilities for start-up messages

Author Peter Ruckdeschel

Maintainer Peter Ruckdeschel <Peter.Ruckdeschel@itwm.fraunhofer.de>

Depends R(>= 1.8.0)

LazyLoad yes

License LGPL-3

LastChangedDate {\$LastChangedDate: 2009-10-16 05:50:41 +0200 (Fr, 16. Okt 2009) \$}

LastChangedRevision {\$LastChangedRevision: 609 \$}

Repository CRAN

Date/Publication 2009-11-03 13:51:14

R topics documented:

myStartupUtilities	2
StartupUtilities	4

Index	8
--------------	----------

myStartupUtilities *Example functions to utilities for start-up messages*

Description

Illustration of package 'startupmsg'

Usage

```
mySMHandler(c)
mystartupMessage(..., domain = NULL, pkg = "", type="version",
                  SMHandler=mySMHandler, endline = FALSE)

buildStartupMessage(..., pkg, library=NULL, domain=NULL,
                    packageHelp=FALSE, MANUAL = NULL, VIGNETTE = NULL,
                    SMHandler=mySMHandler)
```

Arguments

c	an object of class StartupMessage
pkg	a character string with the name of a single package
library	a character vector with path names of R libraries, or NULL. The default value of NULL corresponds to all libraries currently known. If the default is used, the loaded packages are searched before the libraries
domain	see <code>gettext</code> . If NA, messages will not be translated.
type	character – the type part of an S3-object of class StartupMessage; currently, type should be in <code>c("version", "notabene", "information")</code> .
SMHandler	function to handle the output of an object of class StartupMessage, defaults to <code>mySMHandler</code> ; btw: SMHandler stands for /S/tartup/M/essage/Handler/
endline	logical: shall there be an empty line (TRUE) or a line with <code>linestarter</code> in the end?
packageHelp	logical: is there help available as <code>?<pkg-name> ?</code>
MANUAL	character or NULL if <code>!is.null(MANUAL)</code> the name of a manual distributed together with the package (with relative path within the library) or an URL
VIGNETTE	character or NULL if <code>!is.null(VIGNETTE)</code> an indication of one or more vignettes available to this package
...	character vectors (which are pasted together with no separator)

Details

`mystartupMessage` redirects the output of `startupMessage` to have a particular output function `SMHandler` issue the message; to see how such a function may be defined, have a look at code of the default function `mySMHandler`:

```
mySMHandler <- function(c) {
  pkg <- startupPackage(c) # get the package slot of c
  #prefix a starter for each new line of the message:
  linestarterN <- paste(":",pkg,"> ", sep = "")
  linestarterN <- paste("\n",linestarter, sep = "")
  linestarterE <- paste(linestarterN,"$", sep="")
  writeLines(paste(linestarter, sub(linestarterE, "\n",
    gsub("\n",linestarterN,conditionMessage(c))),
    sep=""), stderr()) }
```

Just like for `startupMessage`, for `mystartupMessage`, too, `restarts muffleMessage()`, `onlytypeMessage(c0,atypes)`, `custom(c, f)` are available (confer [startupmsg](#)).

To generate a complete start-up message, we provide `buildStartupMessage`: this function automatically generates

- a start-up message of class `StartupMessage` with type "version" as to the version information.
- additional messages of class `StartupMessage` and of type "notabene" according to the ... argument
- a message of class `StartupMessage` and of type "information" mentioning
 - ?"<pkg-name>" – according to argument `packageHelp`,
 - NEWS ("<pkg-name>"), if there is a 'NEWS' file,
 - URL, if there is a URL mentioned in the 'DESCRIPTION' file,
 - if there is a MANUAL argument, the file / the URL to this manual
 - if there is a VIGNETTE argument, VIGNETTE is printed out indicating a vignette location

The user may suppress the start-up messages produced by `buildStartupMessages` in two ways: Either by `suppressStartupMessages(expr)` and `onlyversionStartupMessages(expr, atypes="version")` as for `startupmessage` (confer [startupmsg](#)), or – as proposed by Brian Ripley – by options; let us describe the latter possibility here:

- `options("StartupBanner"="off")` switches off all start-up messages
- if option "StartupBanner" is not defined (default) or setting `options("StartupBanner"=NULL)` or `options("StartupBanner"="complete")` the complete start-up banner is displayed
- for any other value of option "StartupBanner" (i.e., not in `c(NULL, "off", "complete")`) only the version information is displayed

Author(s)

Peter Ruckdeschel <Peter.Ruckdeschel@itwm.fraunhofer.de>

See Also

Mail "[Rd] Wishlist: 'quietly' argument for `.onAttach()` / `.First.lib()`" on r-devel by Brian Ripley, <https://stat.ethz.ch/pipermail/r-devel/2006-April/037281.html>

Examples

```
## a set of test messages

msg <- "Note that you may set global options by options() --- cf. ?\"options\"."

## issuing of messages controlled by options()
buildStartupMessage(pkg="stats", msg)
suppressStartupMessages(buildStartupMessage(pkg="stats", msg) )
suppressMessages(buildStartupMessage(pkg="stats", msg))
onlytypeStartupMessages(buildStartupMessage(pkg="stats", msg),
                          atypes="version")

getOption("StartupBanner")
buildStartupMessage(pkg="stats", msg)

options("StartupBanner"="off");getOption("StartupBanner")
buildStartupMessage(pkg="stats", msg)

options("StartupBanner"="complete");getOption("StartupBanner")
buildStartupMessage(pkg="stats", msg)

options("StartupBanner"="something else");getOption("StartupBanner")
buildStartupMessage(pkg="stats", msg)

options("StartupBanner"=NULL);getOption("StartupBanner")
buildStartupMessage(pkg="stats", msg)

MNH <- "http://www.r-project.org/"
buildStartupMessage(pkg="stats", msg, packageHelp=TRUE, MANUAL=MNH)
## not quite a manual, but to illustrate the principle:
## "demo/nlm.R" as a "manual": to be system-independent the
## first call is to be preferred
buildStartupMessage(pkg="stats", msg, packageHelp=TRUE, MANUAL=c("demo", "nlm.R"))
### works, too, (i.e. is equivalent) under Linux and Windows (at least):
buildStartupMessage(pkg="stats", msg, packageHelp=TRUE, MANUAL="demo/nlm.R")
```

StartupUtilities *Utilities for start-up messages*

Description

several utilities to produce start-up messages

Usage

```
readVersionInformation(pkg, library=NULL)
readURLInformation(pkg, library=NULL)
pointertoNEWS(pkg, library=NULL)
```

```

infoShow(pkg, filename, library=NULL)
NEWS(pkg, library=NULL)
TOBEDONE(pkg, library=NULL)

StartupMessage(message, call = NULL, pkg="",
               type="version", newline = FALSE)
startupPackage(startupmessage)
startupType(startupmessage)
startupEndline(startupmessage)

startupMessage(..., domain = NULL, pkg = "",
               type="version", newline = FALSE)

suppressStartupMessages(expr)
onlytypeStartupMessages(expr, atypes="version")

```

Arguments

<code>pkg</code>	a character string with the name of a single package
<code>library</code>	a character vector with path names of R libraries, or <code>NULL</code> . The default value of <code>NULL</code> corresponds to all libraries currently known. If the default is used, the loaded packages are searched before the libraries
<code>filename</code>	name of the file which is to be displayed by <code>infoShow</code> (with relative path within the package main folder)
<code>message</code>	a character string – the message part of an S3-object of class <code>StartupMessage</code>
<code>call</code>	a call expression – the call part of an S3-object of class <code>StartupMessage</code>
<code>type</code>	character – the type part of an S3-object of class <code>StartupMessage</code> ; currently, type should be in <code>c("version", "notabene", "information")</code> .
<code>newline</code>	a logical – the decision on the ending of an S3-object of class <code>StartupMessage</code>
<code>startupmessage</code>	the <code>StartupMessage</code> object whose slot <code>package</code> is to be inspected
<code>domain</code>	see <code>gettext</code> . If <code>NA</code> , messages will not be translated.
<code>atypes</code>	a vector of characters – the types of <code>StartupMessage</code> -objects which <code>onlytypeStartupMessages</code> lets pass through
<code>expr</code>	expression to evaluate.
<code>...</code>	character vectors (which are pasted together with no separator)

Details

`readVersionInformation` and `readURLInformation` read the ‘DESCRIPTION’ file of the package. `readVersionInformation` returns a list with elements `ver` and `title` for the version and title to be found in the ‘DESCRIPTION’ file; if there is a URL entry it is returned by `readURLInformation` else `readURLInformation` returns `NULL`.

If there is a ‘NEWS’ in the package main folder, `pointertoNEWS` returns a string with an expression how to retrieve this file from within R, else `pointertoNEWS` returns `NULL`.

`infoShow` displays the file `filename` in the package main folder using `file.show` – if it exists; `NEWS` in particular displays the ‘NEWS’ file, and analogously, `TOBEDONE` in particular displays the ‘TOBEDONE’ file; takes up an idea by Andy Liaw.

A new sub-condition `StartupMessage` to `message` is introduced, with a constructor with the same name.

In addition to the slots of `message`, it also has slots `package` (for the package they are for), `type` (currently in `c("version", "notabene", "information")`), and `endline` (a logical). These slots may be accessed by `startupPackage`, `startupType`, and `startupEndline`, respectively.

`startupMessage` issues a start-up message which also is represented as a condition. While the start-up message is being processed, the following restarts are available:

- `muffleMessage()` to suppress the `StartupMessage`,
- `onlytypeMessage(c0, atypes)` to filter out types not mentioned in `atypes` of `StartupMessages`-argument `c0`,
- `custom(c, f)` to apply the user-defined function `f` to `StartupMessages`-argument `c0` instead of the usual procedure

The user may suppress the start-up messages produced by these utilities as follows:

- `suppressStartupMessages(expr)` suppresses all messages issued by `startupMessage` in the expression `expr` within the parentheses
- `suppressPackageStartupMessages(expr)`: from package version 0.5 on, is the same as `suppressStartupMessages` for our start-up banners, but more generally suppresses all messages of S3-class `packageStartupMessage` (from **base** package)
- `onlyversionStartupMessages(expr, atypes="version")` only shows messages issued by `startupMessage` in the expression `expr` within the parentheses, if there slot `type` is contained in the `atypes` argument
- by the `custom` restart (see example by Seth Falcon)

Acknowledgement

Thanks to Seth Falcon for his helpful comments.

Author(s)

Peter Ruckdeschel (Peter.Ruckdeschel@itwm.fraunhofer.de)

See Also

`buildStartupMessage` for some illustration; for the ideas taken up in this package, see mails "[Rd] Wishlist: 'quietly' argument for `.onAttach()/.First.lib()`" on r-devel by Brian Ripley, <https://stat.ethz.ch/pipermail/r-devel/2006-April/037281.html>, by Andy Liaw, <https://stat.ethz.ch/pipermail/r-devel/2006-April/037286.html>, by Seth Falcon, <https://stat.ethz.ch/pipermail/r-devel/2006-April/037317.html>, and again by Seth Falcon, <https://stat.ethz.ch/pipermail/r-devel/2006-April/037367.html>, and by the author, <https://stat.ethz.ch/pipermail/r-devel/2006-April/037382.html>

Examples

```

## a set of test messages
several.messages<-function() {
  startupMessage("this is a type 'version' startup message", pkg="PKG")
  startupMessage("this is a type 'information' startup message",
                 pkg="PKG", type="information")
  message("this is an ordinary message")}

## issuing of messages with different wrappers
several.messages()
suppressStartupMessages(several.messages())
suppressMessages(several.messages())
onlytypeStartupMessages(several.messages(), atypes=c("version", "notabene"))

##Example by Seth Falcon:
## Here is a test function
doit <- function() {
  several.messages()
  return(123)
}

## Here is an example message handler. Here, you could write messages
## to a file, send them as email to your friends or enemies, etc.
## For the example, we'll just prepend 'MSG:'
msgLogger <- function(m) {
  types<-paste("(", startupType(m), "):", sep="")
  cat(paste("MSG: ", types, conditionMessage(m)), "\n")
}

## Finally, call the doit function and customize how messages are
## handled.
withCallingHandlers(doit(),
                    StartupMessage=function(m) {
                      invokeRestart("custom", m, msgLogger)
                    })

### reading information file utilities
readVersionInformation("stats")
readURLInformation("stats")
## for packages with URL file see e.g. dsel
pointertoNEWS("stats") ## no NEWS file;
NEWS("stats") ## no NEWS file;
## for packages with NEWS file see e.g. randomForest, distr

```

Index

*Topic **utilities**

myStartupUtilities, [2](#)
StartupUtilities, [4](#)

buildStartupMessage, [6](#)
buildStartupMessage
 (*myStartupUtilities*), [2](#)

infoShow (*StartupUtilities*), [4](#)

mySMHandler (*myStartupUtilities*),
 [2](#)

mystartupMessage
 (*myStartupUtilities*), [2](#)
myStartupUtilities, [2](#)

NEWS (*StartupUtilities*), [4](#)

onlytypeStartupMessages
 (*StartupUtilities*), [4](#)

pointertoNEWS (*StartupUtilities*),
 [4](#)

readURLInformation
 (*StartupUtilities*), [4](#)
readVersionInformation
 (*StartupUtilities*), [4](#)

startupEndline
 (*StartupUtilities*), [4](#)
StartupMessage
 (*StartupUtilities*), [4](#)
startupMessage
 (*StartupUtilities*), [4](#)
startupmsg, [3](#)
startupmsg (*StartupUtilities*), [4](#)
startupPackage
 (*StartupUtilities*), [4](#)
startupType (*StartupUtilities*), [4](#)
StartupUtilities, [4](#)

suppressStartupMessages
 (*StartupUtilities*), [4](#)

TOBEDONE (*StartupUtilities*), [4](#)