Package 'SOAR'

July 21, 2025

Version 0.99-11

Date 2013-12-11

Title Memory management in R by delayed assignments

Author Bill Venables, based on original code by David Brahm

Maintainer Bill Venables <Bill.Venables@gmail.com>

Depends R (>= 2.9.0)

Description Allows objects to be stored on disc and automatically recalled into memory, as required, by delayed assignment.

ByteCompile true

License GPL-2 | GPL-3

NeedsCompilation no

Repository CRAN

Date/Publication 2013-12-11 07:32:44

Contents

SOAR-package *SOAR: Utility functions for memory management*

Description

This suite of functions has two distinct purposes:

- To provide an easy mechanism to store objects on the disc, releasing memory during the R session, but in such a way as to keep them visible on the search path and automatically loaded into memory if and when they are needed, and
- To allow objects to be made automatically available to multiple R sessions, possibly for testing prior to including them in formal packages.

Details

Package:	SOAR
Type:	Package
Version:	0.99-11
Date:	2013-12-12
License:	GPL-2 or GPL-3

The function Store is used to take objects from memory and store them as .RData files, usually in a sub-directory of the current working directory we call a *stored object cache*. The directory is automatically created if required. The search path is augmented to contain an entry that mirrors the stored object cache, in the sense that if an object in the cache is required in future it is loaded into memory by the same lazy loading mechanism as is used in packages.

An established stored object cache may be added to, or re-positioned on, the search path by the function Attach, and the function Objects (alias: Ls) may be used to display the objects currently held in a stored object cache.

Objects may be removed from the stored object cache using the function Remove.

Any of the functions Store, Objects or Remove will silently attach existing stored object caches to the search path as required.

Variants on the four basic functions with "Data" or "Utils" in their names, such as StoreUtils or ObjectsData differ from the corresponding basic version only in the way that their default argument values are defined. The intention is to make it possible either to add to a *local* stored object cache or to a *central* stored object cache for data or utulity functions in a simple way.

A function Search is also provided to show items on the search together with their lib.loc directories, as appropriate.

Author(s)

Bill Venables, borrowing heavily from David Brahm's package **g.data**.

Maintainer: Bill Venables, <Bill.Venables@CSIRO.au>.

References

David E. Brahm, (2002) Delayed Data Packages, *R News*, **2**, pp 11–12. (Contains a brief discussion of the antecedent package, **g.data**.)

Examples

```
## change default cache, keeping any previous setting
oldLC <- Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache")
Sys.setenv(R_LOCAL_CACHE=".R_Test")

## generate some dummy data
dummy <- rnorm(100)
mn <- mean(dummy)
va <- var(dummy)

Attach()                            # may give warning

## store it in the stored object cache
Store(dummy, mn, va)
Search()

Attach(pos=3)                       # change to pos=3
Search()

Objects()
Remove(mn, va)
Objects()
Remove(Objects())                   # empty the cache

detach(".R_Test")                   # remove from search path
Sys.setenv(R_LOCAL_CACHE=oldLC)     # restore normal default
```

`Attach` *Attach object cache*

Description

Place a stored object cache one the search path, or change the position of such a cache already on the search path.

Usage

```
Attach(lib = Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache"),
       lib.loc = Sys.getenv("R_LOCAL_LIB_LOC", unset = "."),
       pos = 2, uniquely = TRUE, readonly = FALSE, ...)
AttachData(...)
AttachUtils(...)
```

Arguments

lib	The name of the cache directory from which items are to be removed. May be given as a character string, or as a name, (i.e. without quotes) for convenience. The default is as follows: • For Attach, the value of the R environment variable R_LOCAL_CACHE, or .R_Cache if unset, • For AttachData, the value of the R environment variable R_CENTRAL_DATA, or .R_Data if unset, • For AttachUtils, the value of the R environment variable R_CENTRAL_UTILS, or .R_Utils if unset.
lib.loc	The enclosing directory where the cache directory is to be found. The default is as follows: • For Attach, the value of the R environment variable R_LOCAL_LIB_LOC, or the current working directory if unset, • For AttachDate and AttachUtils the value of the R environment variable R_CENTRAL_LIB_LOC, or he user's HOME directory if unset.
pos	The position on the search path where the object cache is to be placed, or the new position if the cache is already on the search path.
uniquely	Logical. Are multiple copies of the same cache on the search path to be disallowed? If TRUE, a single copy only is left on the search path.
readonly	Logical. If TRUE modifications to the objects in the cache are prevented.
...	Extra arguments to be passed on to internal SOAR functions. Presently unused.

Details

An existing object cache is attached to the search path. If the object cache directory, file.path(lib.loc, lib), currently does not exist a warning is issued to that effect, but also advising that the directory will be created when an object is to be Stored there.

Value

Nothing of interest. The function is used solely for its side-effect on the search path

Note

If the cache is not presently attached to the search path, it is sliently attached at postion 2 before objects are removed from it.

Old caches made in pre-release versions of SOAR (known as ASOR) will be converted to the present format with a warning that this is happening. After this conversion only the current version of the package may be used to access the cached objects.

Note

This function is not often needed, as any of the other main functions, (Store, Objects, Remove and their variants), will automatically attach the object cache if required to do so. A common use is to change the position of a currently attached object cache on the search path.

To release an object cache from the search path, use the standard function detach.

Author(s)

Bill Venables

References

None

See Also

attach, detach.

Examples

```
## change default cache, keeping any previous setting
oldLC <- Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache")
Sys.setenv(R_LOCAL_CACHE=".R_Test")

## generate some dummy data
dummy <- rnorm(100)
mn <- mean(dummy)
va <- var(dummy)

Attach()                            # may give warning

## store it in the stored object cache
Store(dummy, mn, va)
Search()

Attach(pos=3)                       # change to pos=3
Search()

Objects()
Remove(mn, va)
Objects()
Remove(Objects())                   # empty the cache

detach(".R_Test")                   # remove from search path
Sys.setenv(R_LOCAL_CACHE=oldLC)     # restore normal default
```

NAME *Dummy function to keep the package checker quiet.*

Description

Dummy function, please ignore!

Usage

```
NAME()
```

Value

NULL

Examples

```
## This is a dummy function included ONLY to get around
## a limitation of the parser used by the package checker.

## The function is currently defined as
function ()
NULL
```

Objects *List objects in object caches.*

Description

These functions may be used to find stored object caches on the search path and list the objectes stored in them. If the object cache is currently not on the search path it is silently attached at position 2.

Usage

```
Objects(lib = Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache"),
        lib.loc = Sys.getenv("R_LOCAL_LIB_LOC", unset = "."),
        all.names = FALSE, pattern = ".*", readonly = FALSE)
ObjectsData(...)
ObjectsUtils(...)
Ls(lib = Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache"),
        lib.loc = Sys.getenv("R_LOCAL_LIB_LOC", unset = "."),
        all.names = FALSE, pattern = ".*", readonly = FALSE)
LsData(...)
LsUtils(...)
```

Arguments

lib	The name of the cache directory from which items are to be removed. May be given as a character string, or as a name, (i.e. without quotes) for convenience. The default is as follows: • For Objects, the value of the R environment variable R_LOCAL_CACHE, or .R_Cache if unset, • For ObjectsData, the value of the R environment variable R_CENTRAL_DATA, or .R_Data if unset, • For ObjectsUtils, the value of the R environment variable R_CENTRAL_UTILS, or .R_Utils if unset.
lib.loc	The enclosing directory where the cache directory is to be found. The default is as follows:

- For `Objects`, the value of the `R` environment variable `R_LOCAL_LIB_LOC`, or the current working directory if unset,
- For `ObjectsDate` and `ObjectsUtils` the value of the `R` environment variable `R_CENTRAL_LIB_LOC`, or he user's `HOME` directory if unset.

`all.names`	Logical. Should all names be listed? Normally objects with names beginning with a period are not listed.
`pattern`	Regular expression giving the pattern of object names to be listed.
`readonly`	If the stored object cache is not present on the search path, it is silently attached at position 2. Should it be attached as 'read only'?
`...`	Dummy argument to allow any of the above to be specified.

Details

These convenience functions provide the same functionality as the standard function `objects`, or equivalently `ls`, but specialised to stored object caches. They automatically locate caches on the search path and, optionally, attach them if not currently present.

Value

A character string vector of object names.

Note

The standard functions `objects` or `ls` may always be used on stored object caches, but require the position on the search path to be specified.

Author(s)

Bill Venables

References

None

See Also

`objects`, `ls`.

Examples

```
## change default cache, keeping any previous setting
oldLC <- Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache")
Sys.setenv(R_LOCAL_CACHE=".R_Test")

## generate some dummy data
dummy <- rnorm(100)
mn <- mean(dummy)
va <- var(dummy)
```

```
## store it in the stored object cache
Store(dummy, mn, va)
Search()
Objects()
Remove(mn, va)
Objects()
Remove(Objects())                  # empty the cache
detach(".R_Test")                  # remove from search path
Sys.setenv(R_LOCAL_CACHE=oldLC)    # restore normal default
```

Remove *Remove stored objects from the disc.*

Description

These utilities may be used to remove objects under delayed assignment from the disc permanently, where objects are respectively held in a local cache, a central data cache or central utilities cache.

Usage

```
Remove(..., list = character(0),
       lib = Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache"),
       lib.loc = Sys.getenv("R_LOCAL_LIB_LOC", unset = "."))
RemoveData(...)
RemoveUtils(...)
```

Arguments

`...`	items to be removed. Names are taken as objects to be removed. Character strings, or calls resulting in character strings, are taken as providing names of items to be removed as character strings.
`list`	a character string vector providing the names of objects to be removed. An alternative to `...` allowing the user to provide an explicit list of names.
`lib`	The name of the cache directory from which items are to be removed. May be given as a character string, or as a name, (i.e. without quotes) for convenience. The default is as follows: • For `Remove`, the value of the `R` environment variable `R_LOCAL_CACHE`, or `.R_Cache` if unset, • For `RemoveData`, the value of the `R` environment variable `R_CENTRAL_DATA`, or `.R_Data` if unset, • For `RemoveUtils`, the value of the `R` environment variable `R_CENTRAL_UTILS`, or `.R_Utils` if unset.
`lib.loc`	The enclosing directory where the cache directory is to be found. The default is as follows: • For `Remove`, the value of the `R` environment variable `R_LOCAL_LIB_LOC`, or the current working directory if unset, • For `RemoveDate` and `RemoveUtils` the value of the `R` environment variable `R_CENTRAL_LIB_LOC`, or he user's `HOME` directory if unset.

Details

Linking the default values of `lib` and `lib.loc` to environment variables allows the user to re-set the defaults, if need be, either during startup or in the R session.

Value

Nothing. Used only for side-effects.

Note

If the cache is not presently attached to the search path, it is sliently attached at postion 2 before objects are removed from it.

Old caches made in pre-release versions of SOAR (known as ASOR) will be converted to the present format with a warning that this is happening. After this conversion only the current version of the package may be used to access the cached objects.

Author(s)

Bill Venables

References

None

See Also

`rm`, `remove`.

Examples

```
## change default cache, keeping any previous setting
oldLC <- Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache")
Sys.setenv(R_LOCAL_CACHE=".R_Test")

## generate some dummy data
dummy <- rnorm(100)
mn <- mean(dummy)
va <- var(dummy)

## store it in the stored object cache
Store(dummy, mn, va)
Search()
Objects()
Remove(mn, va)
Objects()
Remove(Objects())                   # empty the cache
detach(".R_Test")                   # remove from search path
Sys.setenv(R_LOCAL_CACHE=oldLC)     # restore normal default
```

Search *Slightly enhanced display of the search path*

Description

Produces an object which, when printed, shows each entry on the search path, together with the enclosing directory, or `lib.loc`, where the entry is a package or similar.

Usage

```
Search(abbrev = FALSE)
```

Arguments

`abbrev`	If the `lib.loc` is long should it be truncated to avoid overwhelming the display? • If `FALSE` (the default), no truncation, • If `TRUE` truncate on the left leaving the last 50 characters, • If numeric, truncate on the left leaving the last `max(1,abbrev)` characters.

Details

Provides a way to distinguish between multiple entries on the search path with the same name, by showing their `lib.loc` directories, if any.

Value

A two-column character matrix with the first column showing the names of the entries on the search path and the second their `lib.loc` directories, where applicable. The printed matrix will have quotes suppressed with `noquote`.

Note

May be independently useful outside the `ASOR` package.

Author(s)

Bill Venables

References

None

See Also

`search`.

Examples

```
Search()
```

Store *Store objects out of memory in a stored object cache.*

Description

These functions take objects in memory and store them on the disc in a directory we call a "stored object cache". The objects remain visible and are brought back into memory as required using the same mechanism as is used for lazy loading in packages. If the stored object cache does not already exist it is created. If it is not already attached to the search path it is silently attached.

Usage

```
Store(..., list = character(0),
      lib = Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache"),
      lib.loc = Sys.getenv("R_LOCAL_LIB_LOC", unset = "."),
      remove = TRUE)
StoreData(...)
StoreUtils(...)
```

Arguments

`...`	items to be removed. Names are taken as objects to be removed. Character strings, or calls resulting in character strings, are taken as providing names of items to be removed as character strings.
`list`	a character string vector providing the names of objects to be removed. An alternative to `...` allowing the user to provide an explicit list of names.
`lib`	The name of the cache directory from which items are to be removed. May be given as a character string, or as a name, (i.e. without quotes) for convenience. The default is as follows: • For `Store`, the value of the `R` environment variable `R_LOCAL_CACHE`, or `.R_Cache` if unset, • For `StoreData`, the value of the `R` environment variable `R_CENTRAL_DATA`, or `.R_Data` if unset, • For `StoreUtils`, the value of the `R` environment variable `R_CENTRAL_UTILS`, or `.R_Utils` if unset.
`lib.loc`	The enclosing directory where the cache directory is to be found. The default is as follows: • For `Store`, the value of the `R` environment variable `R_LOCAL_LIB_LOC`, or the current working directory if unset, • For `StoreDate` and `StoreUtils` the value of the `R` environment variable `R_CENTRAL_LIB_LOC`, or he user's `HOME` directory if unset.
`remove`	Logical. Should the objects be removed from the current environment? Normally this would be the case.

Details

These functions take objects currently in memory and store them as `.RData` files in a special directory on the disc, normally a sub-directory of the present working directory. We call the directory a "stored object cache". The objects are then made visible by attaching an environment to the search path which loads the file on demand using essentially the lazy loading technique. The path of the stored object cache is specified in two parts, the `lib.loc` giving the path of the parent directory and the `lib`, giving the name of the directory iteself. This is the same protocol as is used for loading packages using `library` or `require`, for example.

Value

Nothing of interest. Used only for its side effect.

Note

If the cache is not presently attached to the search path, it is sliently attached at postion 2 before objects are removed from it.

Old caches made in pre-release versions of `SOAR` (known as `ASOR`) will be converted to the present format with a warning that this is happening. After this conversion only the current version of the package may be used to access the cached objects.

Author(s)

Bill Venables

References

None.

See Also

`save`, `load`.

Examples

```
## change default cache, keeping any previous setting
oldLC <- Sys.getenv("R_LOCAL_CACHE", unset = ".R_Cache")
Sys.setenv(R_LOCAL_CACHE=".R_Test")

## generate some dummy data
dummy <- rnorm(100)
mn <- mean(dummy)
va <- var(dummy)

## store it in the stored object cache
Store(dummy, mn, va)
Search()
Objects()
Remove(mn, va)
Objects()
```

```
Remove(Objects())                   # empty the cache
detach(".R_Test")                   # remove from search path
Sys.setenv(R_LOCAL_CACHE=oldLC)     # restore normal default
```

Index