

Package ‘R.cache’

April 4, 2012

Version 0.6.2

Date 2012-03-20

Title Fast and light-weight caching (memoization) of objects

Author Henrik Bengtsson <henrikb@braju.com>

Maintainer Henrik Bengtsson <henrikb@braju.com>

Depends R (>= 2.5.0), R.methodsS3 (>= 1.2.2), R.oo (>= 1.9.3), R.utils (>= 1.12.0)

Suggests digest (>= 0.5.1)

Description Methods for memoization, that is, caching arbitrary R objects in persistent memory. Objects can be loaded and saved stratified on a set of hashing objects.

License LGPL (>= 2.1)

URL <http://www.braju.com/R/>

LazyLoad TRUE

Repository CRAN

Date/Publication 2012-04-04 20:17:03

R topics documented:

R.cache-package	2
addMemoization	3
clearCache	4
evalWithMemoization	5
getCacheRootPath	7
loadCache	8
memoizedCall	10
readCacheHeader	11
saveCache	12
setCacheRootPath	13

Index	14
--------------	-----------

R.cache-package

Package R.cache

Description

Methods for memoization, that is, caching arbitrary R objects in persistent memory. Objects can be loaded and saved stratified on a set of hashing objects.

Warning: The Application Programming Interface (API) of the classes and methods in this package may change. Classes and methods are considered either to be stable, or to be in beta or alpha (pre-beta) stage.

Requirements

This package requires the **R.oo** [1] and **R.utils** packages. For automatic generation of cache file-names, which is the most common usage of this package, the CRAN package **digest** is also required.

Installation and updates

To install this package, see <http://www.braju.com/R/>.

To get started

- [loadCache](#), [saveCache](#) Methods for loading and saving objects from and to the cache.
- [getCacheRootPath](#), [setCacheRootPath](#) Methods for getting and setting the directory where cache files are stored.

How to cite this package

Whenever using this package, please cite [1] as

```
@INPROCEEDINGS{BengtssonH_2003,
  author      = {Henrik Bengtsson},
  title       = {The {R.oo} package - Object-Oriented Programming
                with References Using Standard {R} Code},
  booktitle   = {Proceedings of the 3rd International Workshop on
                Distributed Statistical Computing (DSC 2003)},
  year        = {2003},
  editor      = {Kurt Hornik and Friedrich Leisch and Achim Zeileis},
  address     = {Vienna, Austria},
  month       = {March},
  issn        = {1609-395X},
  howpublished = {http://www.ci.tuwien.ac.at/Conferences/DSC-2003/},
}
```

Wishlist

Here is a list of features that would be useful, but which I have too little time to add myself. Contributions are appreciated.

- Add a functionality to identify cache files that are no longer of use. For now, there is an extra header field for arbitrary comments which can be used, but maybe more formal fields are useful, e.g. keywords, user, etc?

If you consider implement some of the above, make sure it is not already implemented by downloading the latest "devel" version!

Related work

See also the **filehash** package, and the `cache()` function in the **Biobase** package of Bioconductor.

License

The releases of this package is licensed under LGPL version 2.1 or newer.

The development code of the packages is under a private licence (where applicable) and patches sent to the author fall under the latter license, but will be, if incorporated, released under the "release" license above.

References

- 1 H. Bengtsson, *The R.oo package - Object-Oriented Programming with References Using Standard R Code*, In Kurt Hornik, Friedrich Leisch and Achim Zeileis, editors, Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003), March 20-22, Vienna, Austria. <http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/>

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

addMemoization	<i>Creates a copy of an existing function such that its results are memoized</i>
----------------	--

Description

Creates a copy of an existing function such that its results are memoized.

Usage

```
## Default S3 method:  
addMemoization(fcn, ...)
```

Arguments

fcn	A function (or the name of a function) that should be copied and have memoization added.
...	Additional arguments for controlling the memoization, i.e. all arguments of memoizedCall() that are not passed to do.call() .

Details

The new function is setup such that the the memoized call is done in the environment of the caller (the parent frame of the function).

Value

Returns a [function](#).

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

The returned function utilized [memoizedCall\(\)](#) internally.

clearCache	<i>Removes all files in a cache file directory</i>
------------	--

Description

Removes all files in a cache file directory.

Usage

```
## Default S3 method:
clearCache(path=getCachePath(...), ..., prompt=TRUE & interactive())
```

Arguments

path	A character string specifying the directory to be cleared. By default, the path is what is returned by getCachePath() which arguments ... are also passed.
...	Arguments passed to getCachePath() , especially argument <code>dirs</code> to specify sub-directories.
prompt	If <code>TRUE</code> , the user will be prompted to confirm that the directory will cleared before files are removed.

Details

If the specified directory does not exists, an exception is thrown. Currently, recursive clearing of subdirectories is *not* supported.

Value

Returns (invisibly) a [character vector](#) of pathnames of the files removed. If no files were removed, `NULL` is returned.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

evalWithMemoization *Evaluates an R expression with memoization*

Description

Evaluates an R expression with memoization such that the same objects are assigned to the current environment and the same result is returned, if any.

Usage

```
evalWithMemoization(expr, key=NULL, ..., envir=parent.frame(), force=FALSE)
```

Arguments

<code>expr</code>	The expression to be evaluated.
<code>key</code>	Additional objects to uniquely identify the evaluation.
<code>...</code>	Additional arguments passed to loadCache() and saveCache() .
<code>envir</code>	The environment in which the expression should be evaluated.
<code>force</code>	If <code>TRUE</code> , existing cached results are ignored.

Value

Returns the value of the evaluated `expr` [expression](#), if any.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

Internally, [eval\(\)](#) is used to evaluate the expression.

Examples

```

for (kk in 1:5) {
  printf("Iteration #d:\n", kk)
  res <- evalWithMemoization({
    cat("Evaluating expression...")
    a <- 1
    b <- 2
    c <- 4
    Sys.sleep(1)
    cat("done\n")
    b
  })
  print(res)

  # Sanity checks
  stopifnot(a == 1 && b == 2 && c == 4)

  # Clean up
  rm(a, b, c)
} # for (kk ...)

## OUTPUTS:
## Iteration #1:
## Evaluating expression...done
## [1] 2
## Iteration #2:
## [1] 2
## Iteration #3:
## [1] 2
## Iteration #4:
## [1] 2
## Iteration #5:
## [1] 2

#####
# WARNING
#####
# If the expression being evaluated depends on
# "input" objects, then these must be specified
# explicitly as "key" objects.
for (ii in 1:2) {
  for (kk in 1:3) {
    printf("Iteration #d:\n", kk)
    res <- evalWithMemoization({
      cat("Evaluating expression...")
      a <- kk
      Sys.sleep(1)
      cat("done\n")
      a
    }, key=list(kk=kk))
  }
}

```

```
print(res)

# Sanity checks
stopifnot(a == kk)

# Clean up
rm(a)
} # for (kk ...)
} # for (ii ...)

## OUTPUTS:
## Iteration #1:
## Evaluating expression...done
## [1] 1
## Iteration #2:
## Evaluating expression...done
## [1] 2
## Iteration #3:
## Evaluating expression...done
## [1] 3
## Iteration #1:
## [1] 1
## Iteration #2:
## [1] 2
## Iteration #3:
## [1] 3
```

getCacheRootPath *Gets the root path to the file cache directory*

Description

Gets the root path to the file cache directory.

Usage

```
## Default S3 method:
getCacheRootPath(defaultPath="~/Rcache", ...)
```

Arguments

defaultPath	The default path, if no user-specified directory has been given.
...	Not used.

Value

Returns the path as a [character](#) string.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

To set the directory where cache files are stored, see `setCacheRootPath()`.

Examples

```
print(getCacheRootPath())
```

loadCache

Loads data from file cache

Description

Loads data from file cache, which is unique for an optional key object.

Usage

```
## Default S3 method:
loadCache(key=NULL, sources=NULL, suffix=".Rcache", removeOldCache=TRUE, pathname=NULL, dirs=NULL, ..
```

Arguments

key	An optional object from which a hexadecimal hash code will be generated and appended to the filename.
sources	Optional source objects. If the cache object has a timestamp older than one of the source objects, it will be ignored and removed.
suffix	A character string to be appended to the end of the filename.
removeOldCache	If <code>TRUE</code> and the cache is older than the sources, the cache file is removed, otherwise not.
pathname	The pathname to the cache file. If specified, arguments <code>key</code> and <code>suffix</code> are ignored. Note that this is only needed in order to read a cache file for which the key is unknown, for instance, in order to investigate an unknown cache file.
dirs	A character vector constituting the path to the cache subdirectory (of the <i>cache root directory</i> as returned by <code>getCacheRootPath()</code>) to be used. If <code>NULL</code> , the path will be the cache root path.
...	Additional argument passed to <code>load()</code> .
onError	A character string specifying what the action is if an exception is thrown.

Details

The hash code calculated from the key object is a 32 characters long hexadecimal MD5 hash code. For more details, see the *digest* package.

Value

Returns an R object or `NULL`, if cache does not exist.

Requirements

To make use of the `key` argument, the *digest* package (available on CRAN) must be installed, otherwise an error is generated. That package is not required when `key=NULL`.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

[saveCache\(\)](#).

Examples

```
simulate <- function(mean, sd) {
  # 1. Try to load cached data, if already generated
  key <- list(mean, sd)
  data <- loadCache(key)
  if (!is.null(data)) {
    cat("Loaded cached data\n")
    return(data)
  }

  # 2. If not available, generate it.
  cat("Generating data from scratch...")
  data <- rnorm(1000, mean=mean, sd=sd)
  Sys.sleep(1)           # Emulate slow algorithm
  cat("ok\n")
  saveCache(data, key=key, comment="simulate()")

  data;
}

data <- simulate(2.3, 3.0)
data <- simulate(2.3, 3.5)
data <- simulate(2.3, 3.0) # Will load cached data

# Clean up
file.remove(findCache(key=list(2.3,3.0)))
file.remove(findCache(key=list(2.3,3.5)))
```

memoizedCall	<i>Calls a function with memoization</i>
--------------	--

Description

Calls a function with memoization, that is, caches the results to be retrieved if the function is called again with the exact same arguments.

Usage

```
## Default S3 method:  
memoizedCall(what, ..., envir=parent.frame(), force=FALSE, sources=NULL, dirs=NULL, verbose=FALSE)
```

Arguments

what	The function to be called, or a character string specifying the name of the function to be called, cf. <code>do.call()</code> .
...	Arguments passed to the function.
envir	The environment in which the function is evaluated.
force	If TRUE , any cached results are ignored, otherwise not.
sources, dirs	Optional arguments passed to <code>loadCache()</code> and <code>saveCache()</code> .
verbose	If TRUE , verbose statements are outputted.

Value

Returns the result of the function call.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

Internally, `loadCache()` is used to load memoized results, if available. If not available, then `do.call()` is used to evaluate the function call, and `saveCache()` is used to save the results to cache.

readCacheHeader	<i>Loads data from file cache</i>
-----------------	-----------------------------------

Description

Loads data from file cache, which is unique for an optional key object.

Usage

```
## Default S3 method:  
readCacheHeader(file, ...)
```

Arguments

file	A filename or a connection .
...	Not used.

Value

Returns a named [list](#) structure with element identifier, version, comment (optional), sources (optional), and timestamp.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

[findCache\(\)](#). [loadCache\(\)](#). [saveCache\(\)](#).

Examples

```
data <- 1:120  
key <- list(some=1, vari=2, ables=3)  
  
saveCache(key=key, data, comment="A simple example of a cached object.")  
  
header <- readCacheHeader(findCache(key=key))  
print(header)  
  
# Clean up  
file.remove(findCache(key=key))
```

saveCache	<i>Saves data to file cache</i>
-----------	---------------------------------

Description

Saves data to file cache, which is unique for an optional key object.

Usage

```
## Default S3 method:
saveCache(object, key=NULL, sources=NULL, suffix=".Rcache", comment=NULL, dirs=NULL, compress=getOpti
```

Arguments

object	The object to be saved to file.
key	An optional object from which a hexadecimal hash code will be generated and appended to the filename.
sources	Source objects used for comparison of timestamps when cache is loaded later.
suffix	A character string to be appended to the end of the filename.
comment	An optional character string written in ASCII at the beginning of the file.
dirs	A character vector constituting the path to the cache subdirectory (of the <i>cache root directory</i> as returned by getCacheRootPath()) to be used. If <code>NULL</code> , the path will be the cache root path.
compress	If <code>TRUE</code> , the cache file will be saved using gzip compression, otherwise not.
...	Additional argument passed to save() .

Value

Returns (invisible) the pathname of the cache file.

Requirements

To make use of the key argument, the *digest* package (available on CRAN) must be installed, otherwise an error is generated. That package is not required when `key=NULL`.

Compression

The `saveCache()` method saves a compressed cache file (with filename extension `*.gz`) if argument `compress` is `TRUE`. The `loadCache()` method locates (via [findCache\(\)](#)) and loads such cache files as well.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

For more details on how the hash code is generated etc, [loadCache\(\)](#).

Examples

```
## Not run: For an example, see ?loadCache
```

setCacheRootPath	<i>Sets the root path to the file cache directory</i>
------------------	---

Description

Sets the root path to the file cache directory. By default, this function will set it to `~/ .Rcache/`.

Usage

```
## Default S3 method:  
setCacheRootPath(path="~/ .Rcache", ...)
```

Arguments

path	The path.
...	Not used.

Value

Returns (invisibly) the old root path.

Author(s)

Henrik Bengtsson (<http://www.braju.com/R/>)

See Also

[getCacheRootPath\(\)](#).

Index

*Topic **IO**

- addMemoization, [3](#)
- clearCache, [4](#)
- evalWithMemoization, [5](#)
- getCacheRootPath, [7](#)
- loadCache, [8](#)
- memoizedCall, [10](#)
- readCacheHeader, [11](#)
- saveCache, [12](#)
- setCacheRootPath, [13](#)

*Topic **package**

- R.cache-package, [2](#)

*Topic **programming**

- addMemoization, [3](#)
- clearCache, [4](#)
- evalWithMemoization, [5](#)
- getCacheRootPath, [7](#)
- loadCache, [8](#)
- memoizedCall, [10](#)
- readCacheHeader, [11](#)
- saveCache, [12](#)
- setCacheRootPath, [13](#)

addMemoization, [3](#)

character, [4](#), [5](#), [7](#), [8](#), [10](#), [12](#)

clearCache, [4](#)

connection, [11](#)

do.call, [4](#), [10](#)

environment, [5](#), [10](#)

eval, [5](#)

evalWithMemoization, [5](#)

expression, [5](#)

findCache, [11](#), [12](#)

function, [4](#), [10](#)

getCachePath, [4](#)

getCacheRootPath, [2](#), [7](#), [8](#), [12](#), [13](#)

list, [11](#)

load, [8](#)

loadCache, [2](#), [5](#), [8](#), [10–13](#)

memoizedCall, [4](#), [10](#)

NULL, [5](#), [8](#), [9](#), [12](#)

R.cache (R.cache-package), [2](#)

R.cache-package, [2](#)

readCacheHeader, [11](#)

save, [12](#)

saveCache, [2](#), [5](#), [9–11](#), [12](#)

setCacheRootPath, [2](#), [8](#), [13](#)

TRUE, [4](#), [5](#), [8](#), [10](#), [12](#)

vector, [5](#), [8](#), [12](#)