

Package ‘cacher’

January 2, 2012

Title Tools for caching and distributing statistical analyses

Version 1.1-1

Date 2010-08-26

Depends R (>= 2.10.0)

Imports utils

Suggests digest, CodeDepends, Rgraphviz

Author Roger D. Peng <rpeng@jhsp.h.edu>

Maintainer Roger D. Peng <rpeng@jhsp.h.edu>

Description This package provides tools for caching statistical analyses in key-value databases which can subsequently be distributed over the web.

License GPL (>= 2)

URL <http://github.com/rdpeng/cacher>

Repository CRAN

Date/Publication 2010-08-27 06:23:41

R topics documented:

cache	2
cacher	2
checkobjects	3
clonecache	4
code	5
deletecache	5
getConfig	6
graphcode	7
loadcache	7
objectcode	8

package	9
runcode	10
showfiles	11
showobjects	11
skipcode	12
sourcefile	13

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

cache	<i>Set/Return cache directory</i>
-------	-----------------------------------

Description

Set/Return the path to the current cache directory

Usage

```
cache()
setcache(dir)
```

Arguments

dir directory name

Value

A string indicating the path to the current cache directory.

Examples

```
setcache("mycachedir")
cache()
```

cacher	<i>Cache statistical analyses</i>
--------	-----------------------------------

Description

Caching statistical analyses and R expressions

Usage

```
cacher(srcfile, cachedir = ".cache", logfile = NULL)
cc(expr, cachedir = ".cache", srcfile = NULL, ...)
```

Arguments

srcfile	character, source file to be cached
cachedir	character, path to cache directory
logfile	character, path to file for log messages
expr	an R expression
...	other arguments passed to cacher

Details

cacher caches the results of evaluating the R expressions in the source file `srcfile`. Contents are cached to the directory indicated by `cachedir` and by default, log messages are written to a log file in the cache directory.

The `cc` function caches individual expressions by writing these expressions to a temporary file and calling `cacher` on that file. The name of the temporary file can be set by using the `srcfile` argument. Otherwise, a random name is chosen.

Value

Nothing useful is returned.

Author(s)

Roger D. Peng <rpeng@jhsph.edu>

Examples

```
## Not run:  
sfile <- system.file("examples", "sample.R", package = "cacher")  
cacher(sfile)  
  
## End(Not run)
```

checkobjects	<i>Verify cached objects</i>
--------------	------------------------------

Description

Verify the integrity of objects cached in the database

Usage

```
checkobjects(num)
```

Arguments

num	code expression sequence number
-----	---------------------------------

Details

If `num` is missing, then all of the objects from all of the expressions are checked. Otherwise, `checkobjects` only checks objects in the expressions specified by `num`.

Value

A list of length equal to the number of code expressions specified. Each element of the list is a logical vector indicating which objects for that expression were verified. In the output, `TRUE` indicates that the object was verified and `FALSE` indicates otherwise.

clonecache	<i>Clone a remote cache</i>
------------	-----------------------------

Description

Clone a remote cache or download a remote cache package

Usage

```
clonecache(origin, cachedir = ".cache", all.files = FALSE, id = NULL)
```

Arguments

<code>origin</code>	character, URL of remote cache
<code>cachedir</code>	character, path to local cache directory in which remote cache will be cloned
<code>all.files</code>	logical, should all database files be downloaded?
<code>id</code>	character, identifier for a cache packages stored in a remote archive

Details

If `clonecache` is given a URL via its `origin` argument, then that cache is cloned and (by default) all of the metadata from the remote cache is copied to the local directory indicated by `cachedir`. If `all.files = TRUE` then the database files are also downloaded.

If `id` is non-NULL, then the `origin` argument is ignored and the package corresponding to the ID is downloaded from a remote archive (as set determined by `getConfig("archive")`).

Value

Nothing is returned; a cache directory is created and metadata are downloaded to it.

code	<i>Show code expressions</i>
------	------------------------------

Description

Show code expressions corresponding to a given source file

Usage

```
code(num = NULL, full = FALSE)
showcode()
```

Arguments

num	expression sequence number
full	logical, should the full expression be shown or a one-line abbreviated version?

Details

These functions can be used to show the expressions for a given source file. By default, code shows all expressions in a file in a one-line abbreviated form along with their expression sequence numbers. To see each expression in its entirety, the argument `full = TRUE` must be set. If any expressions have been marked to be skipped by `skipcode`, those expressions will be annotated with an asterisk.

`showcode` shows the original source file in the pager, which can be useful if one is interested in seeing any comments.

Value

An R expression object containing the expressions shown.

deletecache	<i>Delete the cache</i>
-------------	-------------------------

Description

This function clears the entire cache by deleting the cache directory

Usage

```
deletecache(cachedir = NULL)
```

Arguments

cachedir	the path to the cache directory
----------	---------------------------------

Details

If cachedir is NULL, then the cache directory returned by cache is removed.

Value

The result of unlink is returned.

getConfig	<i>Get and set configuration options</i>
-----------	--

Description

Get and set configuration options

Usage

```
getConfig(name)
setConfig(name, value)
```

Arguments

name	character, name of the option
value	option setting

Details

There are a number of options that cacher checks but some of the important ones for users are:

srcfile the source file currently being cached or explored

cachedir the cache directory

archive URL location of the archive from which cache packages should be downloaded

verbose should more messages be printed to the console? Default is FALSE.

Value

getConfig returns the value of the given option or NULL if the option is not valid; setConfig returns nothing.

See Also

[sourcefile](#), [cache](#)

`graphcode`*Plot graph of variable relationships*

Description

Create a plot of a directed graph showing the relationships between the variables in code file or script.

Usage

```
graphcode(num, ...)
```

Arguments

<code>num</code>	a vector of expression sequence numbers
<code>...</code>	other arguments passed to the <code>plot</code> method

Details

This function depends on the **CodeDepends** (for computing dependencies) and **Rgraphviz** (for plotting) packages in order to work. If they are not installed, an error will be thrown.

Value

Nothing useful is returned.

Author(s)

Roger D. Peng <rpeng@jhsph.edu>

`loadcache`*Load objects from cache*

Description

Load R objects corresponding to a given expression from the cache

Usage

```
loadcache(num, env = parent.frame())
```

Arguments

<code>num</code>	expression sequence number
<code>env</code>	environment into which objects should be loaded

Details

Objects associated with an expression can be lazy-loaded into the environment `env` using `loadcache`. One can obtain a list of which objects are associated with which expressions using the `showobjects` function. If `loadcache` is used to load objects from a remote cache, then the corresponding database files will be downloaded on the object's first access.

Expression sequence numbers can be obtained by calling `code`.

Value

A character vector containing the names of the objects loaded (invisibly).

See Also

[code](#), [showobjects](#)

objectcode

Show dependent R expressions for an object

Description

Given the name of an R object or set of R objects, show the R expressions on which those objects depend and return the indices for those R expressions in the document. Alternatively, evaluate the sequence of R expressions that give rise to an R object.

Usage

```
objectcode(name, num, show = TRUE)
evalobject(name, num, env = parent.frame(), ...)
loadobject(name, num, env = parent.frame())
```

Arguments

<code>name</code>	character vector of names of variables
<code>num</code>	a vector of expression sequence numbers
<code>show</code>	Should the expressions be printed to the screen?
<code>env</code>	environment in which evaluation should occur
<code>...</code>	other arguments passed to <code>runcode</code>

Details

These functions depend on the **CodeDepends** package for computing the dependencies. If it is not installed, an error will be thrown.

The `loadobject` function loads the cached value of the expression in which the object is found. It is possible that the cached value of the expression will contain more objects than just the named object, in which case all objects will be loaded.

Value

objectcode returns a vector of indices indicating the expression sequence numbers for the R expressions. evalobject returns nothing but evaluates R expressions in the specified environment.

Author(s)

Roger D. Peng <rpeng@jhsph.edu>

package	<i>Create a cache package</i>
---------	-------------------------------

Description

Create a cache package for distribution

Usage

```
package(cachedir)
cachepackage(cachedir)
```

Arguments

cachedir character, path to cache directory from which the cache package should be made

Details

The zip utility is needed in order for this function to operate. The cache directory is zipped up and an archive is created. This file can be distributed to others or uploaded to an archive website.

If an archive file of the same already exists, it will be overwritten, with a warning issued.

Value

The name of the archive file created.

Note

The package function is deprecated in favor of the renamed cachepackage function.

`runcode`*Evaluate cached R expressions*

Description

Evaluate and check R expressions in a cached analysis

Usage

```
runcode(num, env = parent.frame(), forceAll = FALSE)
checkcode(num, env = globalenv())
```

Arguments

<code>num</code>	expression sequence number
<code>env</code>	environment in which expressions should be evaluated
<code>forceAll</code>	logical, should we load cached expressions when possible or evaluate all expressions?

Details

`checkcode` will evaluate each R expression in the source file on the user's local machine and compare any resulting outputs to the corresponding objects stored in the cache. If the outputs do not match (in the sense that `all.equal` returns something other than `TRUE`), then a message is printed indicating the failure to verify the output and the messages from `all.equal` are provided to the user. Otherwise, `checkcode` will print `OK` for that expression and move to the next expression.

Value

Neither `runcode` nor `checkcode` return anything useful.

Note

Objects involving character values (such as factors) may be subject to collating rules that are specific to the user's local environment. Therefore, if objects are recreated on a user's local machine and compared with the same object in the cache (which was presumably created on a different machine), `checkcode` may report a verification failure because of the change in locale.

See Also

[code](#), [showcode](#)

showfiles	<i>Show source files</i>
-----------	--------------------------

Description

Show the source files that are available for examination

Usage

```
showfiles()
```

Details

This function can be used to show what source files are available in the cache to be examined by the user. One can switch between different source files by calling the `sourcefile` function.

Value

A character vector of the available source files.

See Also

[sourcefile](#)

showobjects	<i>Show objects for an expression</i>
-------------	---------------------------------------

Description

Show what objects are cached for a given expression

Usage

```
showobjects(num)
```

Arguments

num	expression sequence number
-----	----------------------------

Details

Given an expression sequence number, `showobjects` shows what objects were created (and hence cached) by that expression. These objects can subsequently be loaded into the workspace with `loadcache`. If `num` is a sequence, then a single character vector is returned containing the union of the names of the objects cached.

Expression sequence numbers can be obtained by calling `code`.

Value

A character vector containing the names of the objects

See Also

[loadcache](#), [code](#)

skipcode	<i>Skip evaluation of an expression</i>
----------	---

Description

Specify R expressions whose evaluation should be skipped

Usage

```
skipcode(num, append = TRUE)
```

Arguments

num	expression sequence number, or NULL
append	logical, should we add to the list of expressions to be skipped?

Details

skipcode can be used to force certain expressions to be skipped from evaluation when using the runcode function (for example, if certain external resources are not available). There is a globally maintained list of expressions that will be skipped for a given source file.

If num is NULL, then the list of skipped expressions is cleared.

Value

Nothing useful is returned; the global list of expressions to be skipped is modified.

See Also

[code](#)

sourcefile	<i>Get/set source file</i>
------------	----------------------------

Description

Get or set the current source file for analysis

Usage

```
sourcefile(srcfile = NULL)
```

Arguments

srcfile character string indicating the source file for examination, or NULL

Details

When srcfile is NULL, sourcefile returns a character vector of the currently active source file to be examined. This value affects other functions such as code, runcode, etc. If no source file has been previously set, then sourcefile returns NULL.

If srcfile is a character string, then the currently active source file is set to whatever value of srcfile is given.

Value

A character string indicating the currently active source file or NULL if none has been set.

Index

*Topic **utilities**

- cache, [2](#)
 - acher, [2](#)
 - checkobjects, [3](#)
 - clonocache, [4](#)
 - code, [5](#)
 - deletecache, [5](#)
 - getConfig, [6](#)
 - graphcode, [7](#)
 - loadcache, [7](#)
 - objectcode, [8](#)
 - package, [9](#)
 - runcode, [10](#)
 - showfiles, [11](#)
 - showobjects, [11](#)
 - skipcode, [12](#)
 - sourcefile, [13](#)
-
- cache, [2](#), [6](#)
 - cachepackage (package), [9](#)
 - acher, [2](#)
 - cc (acher), [2](#)
 - checkcode (runcode), [10](#)
 - checkobjects, [3](#)
 - clonocache, [4](#)
 - code, [5](#), [8](#), [10](#), [12](#)
-
- deletecache, [5](#)
-
- evalobject (objectcode), [8](#)
-
- getConfig, [6](#)
 - graphcode, [7](#)
-
- loadcache, [7](#), [12](#)
 - loadobject (objectcode), [8](#)
-
- objectcode, [8](#)
-
- package, [9](#)
-
- runcode, [10](#)
-
- setcache (cache), [2](#)
 - setConfig (getConfig), [6](#)
 - showcode, [10](#)
 - showcode (code), [5](#)
 - showfiles, [11](#)
 - showobjects, [8](#), [11](#)
 - skipcode, [12](#)
 - sourcefile, [6](#), [11](#), [13](#)