Package 'fscache'

May 8, 2026

Title File System Cache

Version 1.0.5

Maintainer Pierrick Roger <pierrick.roger@cea.fr>

Description
Manages a file system cache. Regular files can be moved or copied to the cache folder. Subfolders can be created in order to organize the files. Files can be located inside the cache using a glob function. Text contents can be easily stored in and retrieved from the cache using dedicated functions. It can be used for an application or a package, as a global cache, or as a per-user cache, in which case the standard OS user cache folder will be used (e.g.: on Linux $HOME/.cache/R/my_app_or_pkg_cache_folder).

URL https://gitlab.com/cnrgh/databases/r-fscache

BugReports https://gitlab.com/cnrgh/databases/r-fscache/-/issues

Depends R (>= 4.1)

License AGPL-3

Encoding UTF-8

Suggests covr, knitr, lintr, rmarkdown, roxygen2, stringr, testthat (>= 2.0.0)

Imports R.utils, R6, chk, lgr, lifecycle, stringi, tools

NeedsCompilation no

RoxygenNote 7.3.1

Collate 'fcts.R' 'Cache.R' 'package.R'

VignetteBuilder knitr

Author Pierrick Roger [aut, cre] (ORCID: <https://orcid.org/0000-0001-8177-4873>)

Repository CRAN

Date/Publication 2024-06-02 12:30:02 UTC

Contents

`fscache-package` *fscache: File System Cache*

Description

Manages a file system cache. Regular files can be moved or copied to the cache folder. Sub-folders can be created in order to organize the files. Files can be located inside the cache using a glob function. Text contents can be easily stored in and retrieved from the cache using dedicated functions. It can be used for an application or a package, as a global cache, or as a per-user cache, in which case the standard OS user cache folder will be used (e.g.: on Linux $HOME/.cache/R/my_app_or_pkg_cache_folder).

Details

fscache package.

fscache helps you manage file caching for your application. It creates a user cache folder at standard location, or any place you want. You can then save text contents into new files, load contents from existing files, move or copy files into the cache folder. You have also the possibility to create sub-folders in order to organize your files.

Author(s)

Maintainer: Pierrick Roger <pierrick.roger@cea.fr> (ORCID)

See Also

Cache.

`Cache` *A cache class for handling caching on the file system.*

Description

A cache class for handling caching on the file system.

A cache class for handling caching on the file system.

Details

The purpose of this class is to help managing a user cache folder for an application. Files can be copied or moved into the cache folder. Character values can be saved into files and loaded from files. Sub-folders can be defined. Folders can be listed to see the existing files. Files can be deleted individually or by batch, Whole folders can be deleted, including the main cache folder.

Methods

Public methods:

Method `new()`**:** New instance initializer.

Initializes a `Cache` instance, using a specified folder. Path to the folder can be absolute or relative. When relative, the absolute root folder is either the standard user cache folder or the current working directory, depending on `user` parameter.

Usage:

```
Cache$new(folder, user = TRUE, force = FALSE, create = TRUE)
```

Arguments:

`folder` The path to the wanted cache folder. Either an absolute path, or a relative path that will be resolved immediately into an absolute path.

`user` If set to TRUE and the folder path is a relative path, then the path is resolved relatively to the standard user cache folder (i.e.: we call `tools::R_user_dir(folder, which = 'cache')`). A good and standard practice is to set the `folder` parameter to your package name, using.

`force` If the folder exists, is not empty and is not an fscache folder, fails if force is `FALSE`, and use folder anyway if force is `TRUE`.

`create` If `FALSE`, does not create the cache folder if does not exist already. Used for testing purposes only.

Returns: Nothing.

Examples:

```
# Create a new cache instance.
# Note for the sake of the example we use a temporary directory specified
# as an absolute path, however the usual way to use the cache system is
# to provide a relative path, that will be placed inside the standard
# user cache folder defined by the OS.
cache <- Cache$new(tempdir())

# Erase cache
cache$erase()
```

Method `isReadable()`**:** Tests if the cache system is readable.

Cache reading may be disabled and re-enabled with `setReadable()`, Mainly used for debug purposes.

Usage:

```
Cache$isReadable()
```

Returns: `TRUE` if the cache system is readable, `FALSE` otherwise.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Tests if readable (TRUE by default)
if (cache$isReadable()) {
  print("Cache is readable")
}

# Erase cache
cache$erase()
```

Method `isWritable()`**:** Tests if the cache system is writable.

Cache reading may be disabled and re-enabled with `setWritable()`. Mainly used for debug purposes.

Usage:

```
Cache$isWritable()
```

Returns: TRUE if the cache system is writable, FALSE otherwise.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Tests if writeable (TRUE by default)
if (cache$isWritable()) {
  print("Cache is writeable")
}

# Erase cache
cache$erase()
```

Method `setReadable()`**:** Disables or enable cache reading.
Allows or disallows reading to the cache folder.

Usage:

```
Cache$setReadable(readable)
```

Arguments:

`readable` Set to FALSE to disallow reading and to TRUE to allow it.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Disallow reading
cache$setReadable(FALSE)

# Erase cache
cache$erase()
```

Method `setWritable()`**:** Disables or enable cache writing.
Allows or disallows writing to the cache folder.

Usage:

```
Cache$setWritable(writable)
```

Arguments:

`writable` Set to FALSE to disallow writing and to TRUE to allow it.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Disallow writing
```

```
cache$setWritable(FALSE)

# Erase cache
cache$erase()
```

Method `getFolder()`**:** Gets the path to the main cache folder or a sub-folder.

Returns the absolute path to the main cache folder or a cache sub-folder. By default, the folder is created if it does not exist.

Usage:

```
Cache$getFolder(
  sub_folder = NULL,
  create = TRUE,
  fail = FALSE,
  sub.folder = NULL
)
```

Arguments:

`sub_folder` A sub-folder.

`create` If set to `TRUE` and the folder does not exist, creates it.

`fail` If set to `TRUE`, throws an error if the folder does not exist.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: The path to the cache folder as a character value.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the absolute path to the cache folder
folder <- cache$getFolder()

# Get the absolute path to a cache sub-folder
sub_folder <- cache$getFolder('my_sub_folder')

# Erase cache
cache$erase()
```

Method `hasFolder()`**:** Tests if the cache main folder or a cache sub-folder exists.

Usage:

```
Cache$hasFolder(sub_folder = NULL, sub.folder = NULL)
```

Arguments:

`sub_folder` The sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: TRUE if the folder exists. FALSE otherwise.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Test if a sub-folder exists
if (cache$hasFolder("my_sub_folder")) {
  print("Sub-folder exists.")
}

# Erase cache
cache$erase()
```

Method `getPaths()`**:** Computes paths in the cache folder or a cache sub-folder.
Takes a list of relative paths and resolves them using the cache folder path to a list of absolute paths.

Usage:

```
Cache$getPaths(paths, suffix = NULL, sub_folder = NULL, sub.folder = NULL)
```

Arguments:

`paths` A character vector containing paths.

`suffix` A suffix to add to all paths.

`sub_folder` A sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: A character vector, the same size as `paths`, containing the absolute paths.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the paths a list of filenames should have in the cache folder
paths <- cache$getPaths(c("a.csv", "b.txt"))

# Get paths using a common extension for filenames
paths <- cache$getPaths(c("a", "b"), suffix = ".csv")

# Get paths of files inside a sub-folder
paths <- cache$getPaths(c("a.csv", "b.txt"), sub_folder = "foo")

# Erase cache
cache$erase()
```

Method `globPaths()`**:** Search for files inside the cache folder or one of its subfolders.

Usage:

```
Cache$globPaths(
  suffix = NULL,
  sub_folder = NULL,
  tag_files = FALSE,
```

```
  folders = FALSE,
  tag.files = NULL,
  sub.folder = NULL
)
```

Arguments:

`suffix` The suffix files must have.

`sub_folder` A sub-folder where to search.

`tag_files` If set to `FALSE` (default), exclude the tag files. Otherwise include them in the output.

`folders` If set to `FALSE` (default), exclude the folders. Otherwise include them in the output.

`tag.files` **[Deprecated]** Use tag_files instead.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: A character vector containing paths to existing file matching the criteria.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Get all existing files inside sub-folder foo
paths <- cache$globPaths(sub_folder = "foo")

# Get all existing files with extension ".txt" inside main folder
paths <- cache$globPaths(suffix = ".txt")

# Erase cache
cache$erase()
```

Method `getNbItems()`**:** Gets the number of items contained inside a cache folder.

Counts the number of items (files or folders) contained inside a cache folder. This method does not explore the file system recursively, but only look at the files inside the folder.

Usage:

```
Cache$getNbItems(
  sub_folder = NULL,
  tag_files = FALSE,
  folders = FALSE,
  tag.files = NULL,
  sub.folder = NULL
)
```

Arguments:

`sub_folder` A sub-folder.

`tag_files` If set to `FALSE` (default), do not count the tag files. Otherwise count them.

`folders` If set to `FALSE` (default), do not count the folders. Otherwise count them.

`tag.files` **[Deprecated]** Use tag_files instead.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: The number of items.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the number of files inside sub-folder "foo"
n <- cache$getNbItems("foo")

# Erase cache
cache$erase()
```

Method `pathsExist()`**:** Tests if paths exist inside a cache folder.

Takes a list of relative paths and resolves them using the cache folder path to a list of absolute paths. Tests then if each path points to real object on the file system.

Usage:

```
Cache$pathsExist(paths, suffix = NULL, sub_folder = NULL, sub.folder = NULL)
```

Arguments:

`paths` A character vector containing paths.

`suffix` A suffix to add to all paths.

`sub_folder` A sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: A logical vector, the same size as `paths`, with `TRUE` value if the file exists in the cache, or `FALSE` otherwise.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Test if some files exist in the cache
exits <- cache$pathsExist(c("a", "b"), suffix = ".txt")

# Erase cache
cache$erase()
```

Method `tagExists()`**:** Tests if a tag exists in the cache.

Tags are empty files, without extension, whose name starts and ends with `"__"`. This method tests if some tag file already exist in a cache folder.

Usage:

```
Cache$tagExists(name, sub_folder = NULL, sub.folder = NULL)
```

Arguments:

`name` The name of the tag, without the prefix `"__"` and the suffix `"__"`. It will be automatically converted in uppercase. It can only contains digits, letters and underscore characters.

`sub_folder` A sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: TRUE if the tag exists in the cache. FALSE otherwise.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Test if tag file "downloaded" exists in sub-folder "hmdb"
if (cache$tagExists("downloaded", sub_folder = "hmdb")) {
  print("Tag exists")
}

# Erase cache
cache$erase()
```

Method `writeTag()`**:** Sets a tag into the cache.

Usage:

```
Cache$writeTag(name, sub_folder = NULL, sub.folder = NULL)
```

Arguments:

`name` The name of the tag, without the prefix "__" and the suffix "__". It will be automatically converted in uppercase. It can only contains digits, letters and underscore characters.

`sub_folder` A sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Create tag file "downloaded" in sub-folder "hmdb"
cache$writeTag("downloaded", sub_folder = "hmdb")

# Erase cache
cache$erase()
```

Method `getTmp()`**:** Gets path to the cache system temporary folder.

This temporary folder located inside the cache folder is needed in order to be able to move/rename files into the right cache location. When creating files in the system temporary folder, which may reside on a different partition, moving a file could fail as in the following error: cannot rename file "/tmp/Rtmpld18y7/10182e3a086e7b8a7.tsv" to "/home/pr228844/dev/biodb/cache/comp.csv.file-58e...c4/2e3...a7.tsv", reason "Invalid cross-device link".

When you download a file directly to the disk using for instance `download.file()`, write the destination into this destination folder. When downloaded is complete, move the file using the method `importFiles()`.

Usage:

```
Cache$getTmp()
```

Returns: A string containing the path to the folder.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the cache temporary folder
tmp <- cache$getTmp()

# Erase cache
cache$erase()
```

Method `getSubFolders()`**:** Returns all existing sub-folders.

Usage:

```
Cache$getSubFolders()
```

Returns: A character vector containing all the sub-folders.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the list of sub-folders
sub.folders <- cache$getSubFolders()

# Erase cache
cache$erase()
```

Method `importFiles()`**:** Imports existing files into the cache.

Usage:

```
Cache$importFiles(
  src,
  dst = NULL,
  suffix = NULL,
  sub_folder = NULL,
  action = c("copy", "move"),
  sub.folder = NULL
)
```

Arguments:

`src` A character vector containing paths to files to import.

`dst` A character vector containing destination filenames. The vector must have the length as the `src` vector. If `NULL`, the filenames in `src` will be used.

`suffix` A suffix to add to all destination paths.

`sub_folder` A sub-folder. All files will copied or moved to this sub-folder.

`action` Specifies if files have to be moved or copied into the cache.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Create some files for the example
files <- c("k.txt", "u.csv")
file.create(files)

# Move those files into the cache
cache$importFiles(files, sub_folder = "foo", action = "copy")

# Remove original files
unlink(files)

# Erase cache
cache$erase()
```

Method `saveContents()`**:** Saves contents to files into the cache.

Saves character values into files inside a cache folder.

Usage:

```
Cache$saveContents(
  contents,
  dst,
  suffix = NULL,
  sub_folder = NULL,
  sub.folder = NULL
)
```

Arguments:

`contents` A character vector containing the contents to write.

`dst` A character vector containing destination filenames. The vector must have the length as the `contents` vector.

`suffix` A suffix to add to all destination paths.

`sub_folder` A sub-folder. All files will copied or moved to this sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Create some contents for the example
contents <- c("a", "b", "c")

# Save contents
```

```
cache$saveContents(contents, c("a.txt", "b.txt", "c.txt"))

# Erase cache
cache$erase()
```

Method `loadContents()`**:** Loads contents from files stored into the cache.
Loads character values from cache files.

Usage:

```
Cache$loadContents(paths, suffix = NULL, sub_folder = NULL, sub.folder = NULL)
```

Arguments:

`paths` A character vector containing destination filenames. The vector must have the length as the `contents` vector.

`suffix` A suffix to add to all destination paths.

`sub_folder` A sub-folder. All files will copied or moved to this sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: A character vector , the same size as `paths`, containing the contents of the files. If some file does not exist, a `NA` value is returned.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Create some contents for the example
contents <- c("1", "2", "3")

# Save contents
cache$saveContents(contents, c("a", "b", "c"), suffix = ".txt",
                   sub_folder = "ex2")

# Load contents
contents <- cache$loadContents(c("a", "b", "c"), suffix = ".txt",
                               sub_folder = "ex2")

# Erase cache
cache$erase()
```

Method `delPaths()`**:** Deletes a list of paths inside the cache system.
Takes a list of relative paths, resolves them using the cache folder path to a list of absolute paths, and deletes the corresponding files.

Usage:

```
Cache$delPaths(
  paths = NULL,
  suffix = NULL,
  sub_folder = NULL,
  sub.folder = NULL
)
```

Arguments:

`paths` A character vector containing paths.

`suffix` A suffix to add to all paths.

`sub_folder` A sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Delete some cache files
cache$delPaths(c("a.txt", "b.txt"))

# Erase cache
cache$erase()
```

Method `delFolder()`**:** Deletes all files in a sub-folder.

Deletes a sub-folder and all its content.

Usage:

```
Cache$delFolder(sub_folder, sub.folder = NULL)
```

Arguments:

`sub_folder` A sub-folder.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Delete sub-folder
cache$delFolder("my_sub_folder")

# Erase cache
cache$erase()
```

Method `listFolder()`**:** Lists files present inside a cache folder.

Lists files that exist inside a cache folder. Returns by default the full paths of the found files. It is possible to filter on files suffix, and to extract the basename.

Usage:

```
Cache$listFolder(
  sub_folder = NULL,
  suffix = NULL,
  extract_name = FALSE,
```

```
  remove_suffix = FALSE,
  tag_files = FALSE,
  folders = FALSE,
  extract.name = NULL,
  remove.suffix = NULL,
  tag.files = NULL,
  sub.folder = NULL
)
```

Arguments:

`sub_folder` A sub-folder, or `NULL` for the main folder.

`suffix` A file suffix on which to filter.

`extract_name` If set to `TRUE`, instead of returning the full paths of the files, returns their basenames.

`remove_suffix` When set to `TRUE` and `extract.name` is `TRUE` and `suffix` is not `NULL`, remove the suffix from the returned basenames.

`tag_files` If set to `FALSE` (default), exclude the tag files. Otherwise include them in the output.

`folders` If set to `FALSE` (default), exclude the folders. Otherwise include them in the output.

`extract.name` **[Deprecated]** Use extract_name instead.

`remove.suffix` **[Deprecated]** Use remove_suffix instead.

`tag.files` **[Deprecated]** Use tag_files instead.

`sub.folder` **[Deprecated]** Use sub_folder instead.

Returns: The paths to the found files, or the names of the files if `extract.name` is set to `TRUE`.

Examples:

```
# Create a new cache instance
cache <- Cache$new("my_cache_folder")

# List files in sub-folder
files <- cache$listFolder("my_sub_folder")

# Remove cache folder
cache$erase()
```

Method `print()`**:** Displays information about this object.

Usage:

```
Cache$print()
```

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Print information
print(cache)
```

```
# Erase cache
cache$erase()
```

Method `erase()`**:** Erases the whole cache folder.
Deletes the main cache folder and all its files and sub-folders.

Usage:

```
Cache$erase()
```

Returns: Nothing.

Examples:

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Deletes the whole cache content
cache$erase()

# Erase cache
cache$erase()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Cache$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# Create a new cache instance inside a custom folder
cache <- Cache$new(tempdir())

# Create some contents for the example
contents <- c("a", "b", "c")

# Save contents
cache$saveContents(contents, c("a.txt", "b.txt", "c.txt"),
                   sub_folder = "sub1")

# Get list of files inside folder
files <- cache$listFolder("sub1")

# Delete files
cache$delPaths(c("a.txt", "c.txt"), sub_folder = "sub1")

# Delete whole sub-folder
cache$delFolder("sub1")

# Erase cache
```

```
cache$erase()


## ------------------------------------------------
## Method `Cache$new`
## ------------------------------------------------

# Create a new cache instance.
# Note for the sake of the example we use a temporary directory specified
# as an absolute path, however the usual way to use the cache system is
# to provide a relative path, that will be placed inside the standard
# user cache folder defined by the OS.
cache <- Cache$new(tempdir())

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$isReadable`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Tests if readable (TRUE by default)
if (cache$isReadable()) {
  print("Cache is readable")
}

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$isWritable`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Tests if writeable (TRUE by default)
if (cache$isWritable()) {
  print("Cache is writeable")
}

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$setReadable`
## ------------------------------------------------
```

```
# Create a new cache instance
cache <- Cache$new(tempdir())

# Disallow reading
cache$setReadable(FALSE)

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$setWritable`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Disallow writing
cache$setWritable(FALSE)

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$getFolder`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the absolute path to the cache folder
folder <- cache$getFolder()

# Get the absolute path to a cache sub-folder
sub_folder <- cache$getFolder('my_sub_folder')

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$hasFolder`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Test if a sub-folder exists
if (cache$hasFolder("my_sub_folder")) {
  print("Sub-folder exists.")
}
```

```
# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$getPaths`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the paths a list of filenames should have in the cache folder
paths <- cache$getPaths(c("a.csv", "b.txt"))

# Get paths using a common extension for filenames
paths <- cache$getPaths(c("a", "b"), suffix = ".csv")

# Get paths of files inside a sub-folder
paths <- cache$getPaths(c("a.csv", "b.txt"), sub_folder = "foo")

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$globPaths`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Get all existing files inside sub-folder foo
paths <- cache$globPaths(sub_folder = "foo")

# Get all existing files with extension ".txt" inside main folder
paths <- cache$globPaths(suffix = ".txt")

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$getNbItems`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the number of files inside sub-folder "foo"
n <- cache$getNbItems("foo")

# Erase cache
```

```
cache$erase()


## ------------------------------------------------
## Method `Cache$pathsExist`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Test if some files exist in the cache
exits <- cache$pathsExist(c("a", "b"), suffix = ".txt")

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$tagExists`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Test if tag file "downloaded" exists in sub-folder "hmdb"
if (cache$tagExists("downloaded", sub_folder = "hmdb")) {
  print("Tag exists")
}

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$writeTag`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Create tag file "downloaded" in sub-folder "hmdb"
cache$writeTag("downloaded", sub_folder = "hmdb")

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$getTmp`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())
```

```
# Get the cache temporary folder
tmp <- cache$getTmp()

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$getSubFolders`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Get the list of sub-folders
sub.folders <- cache$getSubFolders()

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$importFiles`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Create some files for the example
files <- c("k.txt", "u.csv")
file.create(files)

# Move those files into the cache
cache$importFiles(files, sub_folder = "foo", action = "copy")

# Remove original files
unlink(files)

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$saveContents`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Create some contents for the example
contents <- c("a", "b", "c")
```

```
# Save contents
cache$saveContents(contents, c("a.txt", "b.txt", "c.txt"))

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$loadContents`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Create some contents for the example
contents <- c("1", "2", "3")

# Save contents
cache$saveContents(contents, c("a", "b", "c"), suffix = ".txt",
                   sub_folder = "ex2")

# Load contents
contents <- cache$loadContents(c("a", "b", "c"), suffix = ".txt",
                               sub_folder = "ex2")

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$delPaths`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Delete some cache files
cache$delPaths(c("a.txt", "b.txt"))

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$delFolder`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Delete sub-folder
cache$delFolder("my_sub_folder")
```

```
# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$listFolder`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new("my_cache_folder")

# List files in sub-folder
files <- cache$listFolder("my_sub_folder")

# Remove cache folder
cache$erase()


## ------------------------------------------------
## Method `Cache$print`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Print information
print(cache)

# Erase cache
cache$erase()


## ------------------------------------------------
## Method `Cache$erase`
## ------------------------------------------------

# Create a new cache instance
cache <- Cache$new(tempdir())

# Deletes the whole cache content
cache$erase()

# Erase cache
cache$erase()
```

---

load_text_content *Load the content of a text file.*

Description

Load the content of a text file, trying to guess file encoding. The content is returned as a single character value.

Usage

```
load_text_content(path, min.confidence = 0)
```

Arguments

path	Path to the file from which to load content.
min.confidence	The minimum confidence value (between 0 and 1)

Value

The content of the file as a single character value, NA_character_ if encoding was not recognized.

Examples

```
# Create a file in Latin-1 encoding
text_file <- tempfile('myfile', fileext = '.txt')
dir.create(dirname(text_file), recursive = TRUE)
x <- iconv("Qui sème le vent récolte la tempête.",
           from = "utf8", to = "latin1")
stringi::stri_write_lines(x, text_file, encoding = "latin1")

# Load its content
content <- fscache::load_text_content(text_file)
content

# Remove file
unlink(text_file)
```

Index