Package 'BiocFileCache'

May 4, 2026

Title Manage Files Across Sessions

Version 3.3.0

Description This package creates a persistent on-disk cache of files that the user can add, update, and retrieve. It is useful for managing resources (such as custom Txdb objects) that are costly or difficult to create, web resources, and data files used across sessions.

Depends R (>= 3.4.0), dbplyr (>= 1.0.0)

Imports methods, stats, utils, dplyr, RSQLite, DBI, filelock, curl, httr2

BugReports https://github.com/Bioconductor/BiocFileCache/issues

DevelopmentURL https://github.com/Bioconductor/BiocFileCache

License Artistic-2.0

Encoding UTF-8

RoxygenNote 7.3.2

biocViews DataImport

VignetteBuilder knitr

Suggests testthat, knitr, BiocStyle, rmarkdown, rtracklayer

git_url https://git.bioconductor.org/packages/BiocFileCache

git_branch devel

git_last_commit 20482da

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-04

Author Lori Shepherd [aut, cre], Martin Morgan [aut]

Maintainer Lori Shepherd <lori.shepherd@roswellpark.org>

Contents

`BFCOption` *BFCOption These functions help get and set an R variable CACHE that controls the default caching location.*

Description

BFCOption These functions help get and set an R variable CACHE that controls the default caching location.

Usage

```
setBFCOption(arg, value)
```

Arguments

`arg`	character(1) option to get or set
`value`	The value to be assigned to the designated option

Details

Currently the only supported option is CACHE. This controls the default location of the BiocFileCache caching directory. By default the value is established by `tools::R_user_dir("BiocFileCache",which="cache` This value can also be defaultly set by using system and global environment variables visible *before* the package is loaded. The variable that should be set if utilized is "BFC_CACHE"

Value

Value of request option or invisible successfully set option

Author(s)

Lori Shepherd

Examples

```
origPath = getBFCOption('CACHE')
setBFCOption('CACHE', "~/.myBFC")
```

BiocFileCache-class *BiocFileCache class*

Description

This class represents the location of files stored on disk. Use the return value to add and retrieve files that persist across sessions.

Usage

```
BiocFileCache(cache = getBFCOption("CACHE"), ask = interactive())

## S4 method for signature 'BiocFileCacheBase'
bfccache(x)

## S4 method for signature 'missing'
bfccache(x)

## S4 method for signature 'BiocFileCacheBase'
length(x)

bfcrid(x)

## S4 method for signature 'missing'
bfcrid(x)

## S4 method for signature 'BiocFileCacheReadOnly'
bfcrid(x)

## S4 method for signature 'BiocFileCache'
bfcrid(x)

## S4 method for signature 'BiocFileCache,character,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'BiocFileCacheReadOnly,character,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'BiocFileCache,missing,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'BiocFileCacheReadOnly,missing,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'BiocFileCacheBase,character,missing'
x[[i, j]]

## S4 replacement method for signature 'BiocFileCache,character,missing,character'
x[[i, j, ...]] <- value

## S4 method for signature 'missing'
```

```
bfcnew(
  x,
  rname,
  rtype = c("relative", "local"),
  ext = NA_character_,
  fname = c("unique", "exact")
)

## S4 method for signature 'BiocFileCache'
bfcnew(
  x,
  rname,
  rtype = c("relative", "local"),
  ext = NA_character_,
  fname = c("unique", "exact")
)

## S4 method for signature 'missing'
bfcadd(
  x,
  rname,
  fpath = rname,
  rtype = c("auto", "relative", "local", "web"),
  action = c("copy", "move", "asis"),
  proxy = "",
  download = TRUE,
  progress = TRUE,
  config = list(),
  ext = NA_character_,
  fname = c("unique", "exact"),
  ...
)

## S4 method for signature 'BiocFileCache'
bfcadd(
  x,
  rname,
  fpath = rname,
  rtype = c("auto", "relative", "local", "web"),
  action = c("copy", "move", "asis"),
  proxy = "",
  download = TRUE,
  progress = TRUE,
  config = list(),
  ext = NA_character_,
  fname = c("unique", "exact"),
  ...
)

## S4 method for signature 'missing'
bfcinfo(x, rids)
```

```
## S4 method for signature 'BiocFileCacheBase'
bfcinfo(x, rids)

## S4 method for signature 'tbl_bfc'
bfcrid(x)

## S4 method for signature 'missing'
bfcpath(x, rids)

## S4 method for signature 'BiocFileCacheBase'
bfcpath(x, rids)

## S4 method for signature 'missing'
bfcrpath(x, rnames, ..., rids, exact = TRUE)

## S4 method for signature 'BiocFileCacheBase'
bfcrpath(x, rnames, ..., rids, exact = TRUE)

## S4 method for signature 'missing'
bfcupdate(x, rids, ...)

## S4 method for signature 'BiocFileCache'
bfcupdate(
  x,
  rids,
  ...,
  rname = NULL,
  rpath = NULL,
  fpath = NULL,
  proxy = "",
  progress = TRUE,
  config = list(),
  ask = TRUE
)

bfcmeta(x, name, ...) <- value

## S4 replacement method for signature 'BiocFileCacheBase'
bfcmeta(x, name, ...) <- value

## S4 method for signature 'missing'
bfcmetaremove(x, name, ...)

## S4 method for signature 'BiocFileCacheBase'
bfcmetaremove(x, name, ...)

## S4 method for signature 'missing'
bfcmetalist(x)

## S4 method for signature 'BiocFileCacheBase'
bfcmetalist(x)
```

```
## S4 method for signature 'missing'
bfcmeta(x, name, ...)

## S4 method for signature 'BiocFileCacheBase'
bfcmeta(x, name, ...)

## S4 method for signature 'missing'
bfcquerycols(x)

## S4 method for signature 'BiocFileCacheBase'
bfcquerycols(x)

## S4 method for signature 'missing'
bfcquery(x, query, field = c("rname", "rpath", "fpath"), ..., exact = FALSE)

## S4 method for signature 'BiocFileCacheBase'
bfcquery(x, query, field = c("rname", "rpath", "fpath"), ..., exact = FALSE)

## S4 method for signature 'missing'
bfccount(x)

## S4 method for signature 'BiocFileCacheBase'
bfccount(x)

## S4 method for signature 'tbl_bfc'
bfccount(x)

## S4 method for signature 'missing'
bfcneedsupdate(x, rids, ..., proxy = "", config = list())

## S4 method for signature 'BiocFileCacheBase'
bfcneedsupdate(x, rids, ..., proxy = "", config = list())

## S4 method for signature 'missing'
bfcdownload(
  x,
  rid,
  proxy = "",
  progress = TRUE,
  config = list(),
  ask = TRUE,
  FUN,
  ...
)

## S4 method for signature 'BiocFileCache'
bfcdownload(
  x,
  rid,
  proxy = "",
  progress = TRUE,
  config = list(),
```

```
  ask = TRUE,
  FUN,
  ...
)

## S4 method for signature 'missing'
bfcremove(x, rids)

## S4 method for signature 'BiocFileCache'
bfcremove(x, rids)

## S4 method for signature 'missing'
bfcsync(x, verbose = TRUE, ask = TRUE)

## S4 method for signature 'BiocFileCache'
bfcsync(x, verbose = TRUE, ask = TRUE)

## S4 method for signature 'missing'
exportbfc(
  x,
  rids,
  outputFile = "BiocFileCacheExport.tar",
  outputMethod = c("tar", "zip"),
  verbose = TRUE,
  ...
)

## S4 method for signature 'BiocFileCacheBase'
exportbfc(
  x,
  rids,
  outputFile = "BiocFileCacheExport.tar",
  outputMethod = c("tar", "zip"),
  verbose = TRUE,
  ...
)

## S4 method for signature 'character'
importbfc(filename, archiveMethod = c("untar", "unzip"), exdir = ".", ...)

## S4 method for signature 'missing'
cleanbfc(x, days = 120, ask = TRUE)

## S4 method for signature 'BiocFileCache'
cleanbfc(x, days = 120, ask = TRUE)

## S4 method for signature 'missing'
removebfc(x, ask = TRUE)

## S4 method for signature 'BiocFileCache'
removebfc(x, ask = TRUE)
```

```
## S4 method for signature 'BiocFileCacheBase'
show(object)
```

Arguments

cache	character(1) On-disk location (directory path) of cache. For default location see R_user_dir.
ask	logical(1) Ask before creating, updating, overwriting, or removing cache or local file locations.
x	A BiocFileCache instance or, if missing, the result of BiocFileCache().
i	character() 'rid' identifiers.
j	Ignored.
...	For 'bfcadd', 'bfcupdate' and 'bfcdownload': Additional arguments passed to internal download functions for use with httr2::req_perform. For 'bfcrpaths': Additional arguments passed to 'bfcadd', or exact passed to 'bfcquery'. For 'bfcquery': Additional arguments passed to grepl. For 'exportbfc': Additional arguments to the selected outputMethod function. See utils::tar or utils::zip for more information. For 'importbfc': Additional arguments to the selected archiveMethod function. See utils::untar or utils::unzip for more information.
drop	Ignored.
value	character(1) Replacement file path.
rname	character(1) Name of object in file cache. For 'bfcupdate' a character vector of replacement rnames.
rtype	character(1) 'local', 'relative', or 'web' indicating if the resource is a local file, a relative path in the cache, or a web resource. For bfcnew: local or relative are only options. For bfcadd, the default 'auto' creates relative or web paths, based on the path prefix.
ext	character(1) A file extension to add to the local copy of the file (e.g., 'sqlite', 'txt', 'tar.gz').
fname	character(1). Options are 'unique' or 'exact'. 'unique' provides each bfc resource with a unique identifier when storing the file, allowing resources with the same name to be stored in the cache. 'exact' uses the exact file name of the resource; only one of foo/my.txt and bar/my.txt could be stored. Default is 'unique'.
fpath	For bfcadd(), character(1) path to current file location or remote web resource. If none is given, the rname is assumed to also be the path location. For bfcupdate() character() vector of replacement web resources.
action	character(1) How to handle the file: create a copy of fpath in the cache directory; move the file to the cache directory; or asis leave the file in current location but save the path in the cache. If 'rtype == "relative"', action can not be "asis".
proxy	character(1) (Optional) proxy server passed to httr2::req_proxy
download	logical(1) If rtype=web, should remote resource be downloaded locally immediately.
progress	TRUE/FALSE if progress bar for downloads in interactive session should be shown

config	list() passed as argument to httr2::req_options. The names of items should be valid curl options as defined in curl::curl_options.
rids	character() Vector of rids.
rnames	character() to match against rnames. Each element of rnames must match exactly one record. Use exact = FALSE to use regular expression matching.
exact	logical(1) when FALSE, treat query as a regular expression. When TRUE, use exact matching. For bfcquery, the default is FALSE (regular expression matching; for bfcrpath, the default is TRUE (exact matching).
rpath	character() vector of replacement rpaths.
name	character(1) name of metadata table.
query	character() Regular expression pattern(s) to match in resource. It will match the pattern against fields, using & logic across query element. By default, case sensitive. When exact = TRUE, query uses exact matching.
field	character() column names in resource to query, using || logic across multiple field elements. By default, matches pattern agains rname, rpath, and fpath. If exact matching, may only be a single value.
rid	character(1) Unique resource id.
FUN	A specialized implemented function designed by the user. This function can be used to perform and save the results of a post download processing step rather than direct output. The function should ONLY take in two file names: the first the raw downloaded file and the second the output file for saved results. The output of the function should be TRUE/FALSE if step was successful. See vignette section on Specialty Advance Use Case for more details.
verbose	logical(1) If descriptive message and list of issues should be included as output.
outputFile	character(1) The <filepath>/basename for the output archive. Please include appropriate extension based on outMethod and any additional parameters selected for utils::tar or utils::zip
outputMethod	Either 'tar' or 'zip' for how the directory should be archived. Default is 'tar'.
filename	character(1) The name of the archive.
archiveMethod	Either 'untar' or 'unzip' for how the directory should be extracted. Default is 'untar'.
exdir	Directory to extract files too. See utils::untar or utils::unzip for more details.
days	integer(1) Number of days between accessDate and currentDate; if exceeded entry will be deleted.
object	A BiocFileCache instance.

Details

The package defines 'BiocFileCache', 'BiocFileCacheBase' and 'BiocFileCacheReadOnly' classes.

Slots unique to 'BiocFileCache' and related classes:

'cache': character(1) on-disk location (directory path) of the cache

'rid': character() of unique rids in the cache.

The cache creates an RSQLite database to keep track of local and remote resources. Each item located in the database will have the following information:

'rid': resource id. Autogenerated. This is a unique identifier automatically generated when a resource is added to the cache

'rname': resource name. This is given by the user when a resource is added to the cache. It does not have to be unique and can be updated at anytime. We recommend descriptive key words and identifers.

'create_time': The date and time a resource is added to the cache.

'access_time': The date and time a resource is utilized within the cache. The access time is updated when the resource is updated or accessed

'rpath': resource path. This is the path to the local (on-disk) file

'rtype': resource type. Either "relative", "local", or "web", indicating if the resource has a remote origin

'fpath': If rtype is "web", this is the link to the remote resource. It will be utilized to download or update the remote data

'last_modified_time': For a remote resource, the last_modified (if available) information for the local copy of the data. This information is checked against the remote resource to determine if the local copy is stale and needs to be updated

All functions have a quick implementation where if the BiocFileCache object is not passed as an argument, the function uses default 'BiocFileCache()' for implementation. e.g 'bfcinfo()' can be used instead of 'bfcinfo(BiocFileCache())'. The only function this is not available for is 'bfcmeta()<-'; The BiocFileCache object must be defined as a varaible and passed as an argument. See vignette("BiocFileCache") for more details.

Value

For 'BiocFileCache': a `BiocFileCache` instance.

For 'bfccache': character(1) location of the directory containing the cache.

For 'length': integer(1) Number of objects in the file cache.

For '[': A subset of the BiocFileCache object.

For '[[': named character(1) rpath for the given resource in the cache.

For '[[<-': Updated BiocFileCache, invisibly.

For 'bfcnew': named character(1), the path to save your object / file. The name of the return value is the unique rid for the resource.

For 'bfcadd': named character(1), the path to save your object / file. The name of the character is the unique rid for the resource.

For 'bfcinfo': A `bfc_tbl` of current resources in the database.

For 'bfcpath': the file path location to load

For 'bfcrpath': The local file path location to load.

For 'bfcupdate': an updated `BiocFileCache` object, invisibly.

For 'bfcmeta': updated BiocFileCache, invisibly

For 'bfcmetaremove': updated BiocFileCache, invisibly

For 'bfcmetalist': returns a character() of all metadata tables currently in the database. If no metadata tables are available returns character(0)

For 'bfcmeta': returns a data.frame representation of database table

For 'bfcquerycols': character() all columns in all database tables available for query.

For 'bfcquery': A `bfc_tbl` of current resources in the database whose `field` contained query. If multiple values are given, the resource must contain all of the patterns. A tbl with zero rows is returned when no resources match the query.

For 'bfccount': integer(1) Number of objects in the cache or query.

For 'bfcneedsupdate': named logical vector if resource needs to be updated. The name is the resource 'rid'. `TRUE`: fpath `etag` or `modified` time of web resource more recent than in BiocFileCache; `FALSE`: fpath `etag` or `modified` time of web resource not more recent than in BiocFileCache; `NA`: web resource etag and modified time could not be determined. If the etag is available the function will use that information definitively and only compare last modified time if etag is not available. If there is an `expires` time that will be used to initially determine if the resource should be updated.

For 'bfcdownload': character(1) path to downloaded resource in cache.

For 'bfcremove': updated BiocFileCache object, invisibly

For 'bfcsync': logical(1) indicating whether the cache is in sync (`TRUE`) or not. 'verbose' is TRUE by default, so descriptive messages will also be included.

character(1) The outputFile path.

A BiocFileCache object

For 'cleanbfc': updated BiocFileCache, invisibly.

For 'removebfc': TRUE if successfully removed.

Methods (by generic)

- `bfccache(BiocFileCacheBase)`: Get the location of the on-disk cache.
- `length(BiocFileCacheBase)`: Get the number of objects in the file cache.
- `bfcrid(BiocFileCacheReadOnly)`: Get the rids of the object.
- `x[i`: Subset a BiocFileCache object.
- `x[[i`: Get a file path for select resources from the cache.
- `` `[[`(x = BiocFileCache, i = character, j = missing) <- value ``: Set the file path of selected resources from the cache.
- `bfcnew(BiocFileCache)`: Add a resource to the database
- `bfcadd(BiocFileCache)`: Add an existing resource to the database
- `bfcinfo(BiocFileCacheBase)`: list resources in database
- `bfcrid(tbl_bfc)`: Get the rids of the object
- `bfcpath(BiocFileCacheBase)`: display rpaths of resource.
- `bfcrpath(BiocFileCacheBase)`: display rpath of resource. If 'rnames' is in the cache the path is returned, if it is not it will try to add it to the cache with 'bfcadd'
- `bfcupdate(BiocFileCache)`: Update a resource in the cache
- `bfcmeta(BiocFileCacheBase) <- value`: add meta data table in database
- `bfcmetaremove(BiocFileCacheBase)`: remove meta data table in database
- `bfcmetalist(BiocFileCacheBase)`: retrieve listing of metadata tables
- `bfcmeta(BiocFileCacheBase)`: retrieve metadata table
- `bfcquerycols(BiocFileCacheBase)`: Get all the possible columns to query
- `bfcquery(BiocFileCacheBase)`: query resource
- `bfccount(BiocFileCacheBase)`: Get the number of objects in the file cache or query.

- `bfcneedsupdate(BiocFileCacheBase)`: check if a resource needs to be updated
- `bfcdownload(BiocFileCache)`: Redownload resource to location in cache
- `bfcremove(BiocFileCache)`: Remove a resource to the database. If the local file is located in `bfccache(x)`, the file will also be deleted. This will not delete information in any metadata table.
- `bfcsync(BiocFileCache)`: sync cache and resource.
- `exportbfc(BiocFileCacheBase)`: Create exportable file containing BiocFileCache.
- `importbfc(character)`: Import file created with exportbfc containing BiocFileCache.
- `cleanbfc(BiocFileCache)`: Remove old/unused files in BiocFileCache. If file to be removed is not in the bfccache location it will not be deleted. Setting `days=-Inf` will remove all cached files.
- `removebfc(BiocFileCache)`: Completely remove the BiocFileCache
- `show(BiocFileCacheBase)`: Display a `BiocFileCache` instance.

Examples

```
# bfc <- BiocFileCache()            # global cache
# bfc
bfc0 <- BiocFileCache(tempfile())         # temporary catch for examples
bfccache(bfc0)
length(bfc0)
path <- bfcnew(bfc0, "NewResource")
path
fl1 <- tempfile(); file.create(fl1)
bfcadd(bfc0, "Test1", fl1)                 # copy
fl2 <- tempfile(); file.create(fl2)
bfcadd(bfc0, "Test2", fl2, action="move")         # move
fl3 <- tempfile(); file.create(fl3)
add3 <- bfcadd(bfc0, "Test3", fl3, rtype="local", action="asis")  # reference
rid3 <- names(add3)

bfc0
file.exists(fl1)                                  # TRUE
file.exists(fl2)                                  # FALSE
file.exists(fl3)                                  # TRUE

# add a remote resource
url <- "https://httpbin.org/get"
bfcadd(bfc0, "TestWeb", fpath=url)
bfcinfo(bfc0)
bfcpath(bfc0, rid3)
bfcrpath(bfc0, rids = rid3)
bfcupdate(bfc0, rid3, rpath=fl3, rname="NewRname")
bfc0[[rid3]] = fl1
bfcupdate(bfc0, "BFC5", fpath="http://google.com")
meta = data.frame(list(rid = paste("BFC",seq_len(bfccount(bfc0)), sep=""),
                       num=seq(bfccount(bfc0),1,-1),
                       data=c(paste("Letter",
                       letters[seq_len(bfccount(bfc0))]))),
                  stringsAsFactors=FALSE)
bfcmeta(bfc0, name="resourcedata") <- meta
## Not run: bfcmetaremove(bfc0, "resourcedata")
bfcmetalist(bfc0)
```

```
tbl = bfcmeta(bfc0, "resourcedata")
tbl
bfcquerycols(bfc0)
bfcquery(bfc0, "Test")
bfcquery(bfc0, "^Test1$", field="rname")
bfccount(bfc0)
bfccount(bfcquery(bfc0, "test"))
bfcneedsupdate(bfc0, "BFC5")
bfcdownload(bfc0, "BFC5")
bfcremove(bfc0, rid3)
bfcinfo(bfc0)
bfcsync(bfc0)

if (!interactive()){
   # in interactive mode, in the sync above
   # this was probably already removed
   # noninteractive mode does not remove resources
   # so can remove manually here
   bfcremove(bfc0, "BFC1")
}
bfcsync(bfc0, FALSE)
## Not run: exportbfc(bfc)
## Not run: importbfc("ExportBiocFileCache.tar")
## Not run: cleanbfc(bfc, ask=FALSE)
## Not run: removebfc(bfc, ask=FALSE)
```

`makeBiocFileCacheFromDataFrame` *Make BiocFileCache objects from an existing data.frame*

Description

If there are a lot of resources being added this could take some time but if a cache is saved in a permanent location this should only have to be run once. The original data.frame must have the required columns 'rtype', 'fpath', and 'rpath'; See the vignette for more information on the expected information contained in these columns. Similarly, the optional columns 'rname', 'etag', 'last_modified_time', and 'expires' may be included. Any additional columns not listed as required or optional will be kept as an additional metadata table in the BiocFileCache database.

Usage

```
makeBiocFileCacheFromDataFrame(
  df,
  cache,
  actionLocal = c("move", "copy", "asis"),
  actionWeb = c("move", "copy"),
  metadataName,
  ...,
  ask = TRUE
)

## S4 method for signature 'ANY'
makeBiocFileCacheFromDataFrame(
```

```
  df,
  cache,
  actionLocal = c("move", "copy", "asis"),
  actionWeb = c("move", "copy"),
  metadataName,
  ...,
  ask = TRUE
)
```

Arguments

df	data.frame or tibble to convert
cache	character(1) On-disk location (directory path) of cache. For default location see R_user_dir.
actionLocal	If local copy of file should be moved, copied or left in original location. See 'action' param of bfcadd.
actionWeb	If a local copy of a remote resource already exists, should the file be copied or moved to the cache. Locally downloaded remote resources must exist in the cache location.
metadataName	If there are additional columns of data in the original data.frame besides required BiocFileCache columns, this data will be added as a metadata table with this name.
...	additional arguments passed to 'file.copy()'.
ask	logical(1) Confirm creation of BiocFileCache.

Value

A BiocFileCache object

Examples

```
## Create a data.frame with the required columns
tempf <- tempfile(fileext = ".txt")
file.create(tempf)
df <- data.frame(
    rtype = "local",
    fpath = tempf,
    rpath = tempf,
    rname = "Example File 1",
    etag = "etag1",
    last_modified_time = as.character(Sys.Date()),
    expires = as.character(Sys.Date() + 1),
    stringsAsFactors = FALSE
)
## Create a BiocFileCache from the data.frame
bfc <- makeBiocFileCacheFromDataFrame(
    df, cache = tempfile(), actionLocal = "move",
    metadataName = "resourceMetadata", ask = FALSE
)
bfc
bfcinfo(bfc)
removebfc(bfc, ask = FALSE)
```

makeCachedActiveBinding *makeCachedActiveBinding*

Description

Like makeActiveBinding but the value of the active binding gets only evaluated once and is "remembered".

Usage

```
makeCachedActiveBinding(sym, fun, env = .GlobalEnv, verbose = FALSE)
```

Arguments

sym	See makeActiveBinding in the **base** package.
fun	See makeActiveBinding in the **base** package.
env	See makeActiveBinding in the **base** package.
verbose	Set to TRUE to see caching in action (useful for troubleshooting).

Examples

```
makeCachedActiveBinding("x", function() runif(1), verbose=TRUE)
x
x
```

Index