Package 'archive'

December 11, 2023

Title Multi-Format Archive and Compression Support

Version 1.1.7

Description Bindings to 'libarchive' <http://www.libarchive.org> the Multi-format archive and compression library. Offers R connections and direct extraction for many archive formats including 'tar', 'ZIP', '7-zip', 'RAR', 'CAB' and compression formats including 'gzip', 'bzip2', 'compress', 'lzma' and 'xz'.

License MIT + file LICENSE

URL https://archive.r-lib.org/, https://github.com/r-lib/archive

BugReports https://github.com/r-lib/archive/issues

Depends R (>= 3.6.0)

Imports cli,
glue,
rlang,
tibble

Suggests covr,
testthat

LinkingTo cli,
cpp11

ByteCompile true

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

SystemRequirements libarchive: libarchive-dev (deb), libarchive-devel (rpm), libarchive (homebrew), libarchive_dev (csw)

Biarch true

R topics documented:

`archive` *Construct a new archive*

Description

This function retrieves metadata about files in an archive, it can be passed to `archive_read()` or archive_write to create a connection to read or write a specific file from the archive.

Usage

```
archive(file, options = character())
```

Arguments

`file`	File path to the archive.
`options`	`character()` default: `character(0)` Options to pass to the filter or format. The list of available options are documented in options can have one of the following forms: • `option=value` The option/value pair will be provided to every module. Modules that do not accept an option with this name will ignore it. • `option` The option will be provided to every module with a value of "1". • `!option` The option will be provided to every module with a NULL value. • `module:option=value`, `module:option`, `module:!option` As above, but the corresponding option and value will be provided only to modules whose name matches module. See read options for available read options See write options for available write options

Value

A tibble with details about files in the archive.

See Also

`archive_read()`, `archive_write()` to read and write archive files using R connections, `archive_extract()`, `archive_write_files()`, `archive_write_dir()` to add or extract files from an archive.

Examples

```
a <- archive(system.file(package = "archive", "extdata", "data.zip"))
a
```

`archive_extract` *Extract contents of an archive to a directory*

Description

Extract contents of an archive to a directory

Usage

```
archive_extract(
  archive,
  dir = ".",
  files = NULL,
  options = character(),
  strip_components = 0L
)
```

Arguments

`archive`	`character(1)` The archive filename or an `archive` object.
`dir`	`character(1)` Directory location to extract archive contents, will be created if it does not exist.
`files`	`character() || integer() || NULL` One or more files within the archive, specified either by filename or by position.
`options`	`character()` default: `character(0)` Options to pass to the filter or format. The list of available options are documented in options can have one of the following forms: • `option=value` The option/value pair will be provided to every module. Modules that do not accept an option with this name will ignore it. • `option` The option will be provided to every module with a value of "1". • `!option` The option will be provided to every module with a NULL value. • `module:option=value`, `module:option`, `module:!option` As above, but the corresponding option and value will be provided only to modules whose name matches module. See read options for available read options See write options for available write options
`strip_components`	Remove the specified number of leading path elements. Pathnames with fewer elements will be silently skipped.

Details

If `files` is `NULL` (the default) all files will be extracted.

Value

The filenames extracted (invisibly).

Examples

```
a <- system.file(package = "archive", "extdata", "data.zip")
d <- tempfile()

# When called with default arguments extracts all files in the archive.
archive_extract(a, d)
list.files(d)
unlink(d)

# Can also specify one or more files to extract
d <- tempfile()
archive_extract(a, d, c("iris.csv", "airquality.csv"))
list.files(d)
unlink(d)
```

`archive_read` *Create a readable connection to a file in an archive.*

Description

Create a readable connection to a file in an archive.

Usage

```
archive_read(
  archive,
  file = 1L,
  mode = "r",
  format = NULL,
  filter = NULL,
  options = character()
)
```

Arguments

`archive`	`character(1)` The archive filename or an `archive` object.
`file`	`character(1)` `||` `integer(1)` The filename within the archive, specified either by filename or by position.
`mode`	`character(1)` A description of how to open the connection (if it should be opened initially). See section 'Modes' in `base::connections()` for possible values.
`format`	`character(1)` default: `NULL` The archive format, one of '7zip', 'cab', 'cpio', 'iso9660', 'lha', 'mtree', 'shar', 'rar', 'raw', 'tar', 'xar', 'zip', 'warc'.
`filter`	`character(1)` default: `NULL` The archive filter, one of 'none', 'gzip', 'bzip2', 'compress', 'lzma', 'xz', 'uuencode', 'lzip', 'lrzip', 'lzop', 'grzip', 'lz4', 'zstd'.
`options`	`character()` default: `character(0)` Options to pass to the filter or format. The list of available options are documented in options can have one of the following forms: • `option=value` The option/value pair will be provided to every module. Modules that do not accept an option with this name will ignore it.

- `option` The option will be provided to every module with a value of "1".
- `!option` The option will be provided to every module with a NULL value.
- `module:option=value`, `module:option`, `module:!option` As above, but the corresponding option and value will be provided only to modules whose name matches module. See read options for available read options See write options for available write options

Value

An 'archive_read' connection to the file within the archive to be read.

Examples

```
a <- system.file(package = "archive", "extdata", "data.zip")
# Show files in archive
a

# By default reads the first file in the archive.
read.csv(archive_read(a), nrows = 3)

# Can also specify a filename directly
read.csv(archive_read(a, "mtcars.csv"), nrows = 3)

# Or by position
read.csv(archive_read(a, 3), nrows = 3)

# Explicitly specify the format and filter if automatic detection fails.
read.csv(archive_read(a, format = "zip"), nrows = 3)
```

`archive_write` *Create a writable connection to a file in an archive.*

Description

Create a writable connection to a file in an archive.

Usage

```
archive_write(
  archive,
  file,
  mode = "w",
  format = NULL,
  filter = NULL,
  options = character()
)
```

Arguments

`archive`	`character(1)` The archive filename or an `archive` object.
`file`	`character(1) || integer(1)` The filename within the archive, specified either by filename or by position.

mode — character(1) A description of how to open the connection (if it should be opened initially). See section 'Modes' in base::connections() for possible values.

format — character(1) default: NULL The archive format, one of '7zip', 'cab', 'cpio', 'iso9660', 'lha', 'mtree', 'shar', 'rar', 'raw', 'tar', 'xar', 'zip', 'warc'.

filter — character(1) default: NULL The archive filter, one of 'none', 'gzip', 'bzip2', 'compress', 'lzma', 'xz', 'uuencode', 'lzip', 'lrzip', 'lzop', 'grzip', 'lz4', 'zstd'.

options — character() default: character(0) Options to pass to the filter or format. The list of available options are documented in options can have one of the following forms:

- option=value The option/value pair will be provided to every module. Modules that do not accept an option with this name will ignore it.
- option The option will be provided to every module with a value of "1".
- !option The option will be provided to every module with a NULL value.
- module:option=value, module:option, module:!option As above, but the corresponding option and value will be provided only to modules whose name matches module. See read options for available read options See write options for available write options

Details

If format and filter are NULL, they will be set automatically based on the file extension given in file when writing and automatically detected using Robust automatic format detection when reading.

For traditional zip archives archive_write() creates a connection which writes the data to the specified file directly. For other archive formats the file size must be known when the archive is created, so the data is first written to a scratch file on disk and then added to the archive. This scratch file is automatically removed when writing is complete.

Value

An 'archive_write' connection to the file within the archive to be written.

Examples

```
# Archive format and filters can be set automatically from the file extensions.
f1 <- tempfile(fileext = ".tar.gz")

write.csv(mtcars, archive_write(f1, "mtcars.csv"))
archive(f1)
unlink(f1)

# They can also be specified explicitly
f2 <- tempfile()
write.csv(mtcars, archive_write(f2, "mtcars.csv", format = "tar", filter = "bzip2"))
archive(f2)
unlink(f2)

# You can also pass additional options to control things like compression level
f3 <- tempfile(fileext = ".tar.gz")
write.csv(mtcars, archive_write(f3, "mtcars.csv", options = "compression-level=2"))
archive(f3)
unlink(f3)
```

archive_write_dir *Add files to a new archive*

Description

`archive_write_files()` adds one or more files to a new archive. `archive_write_dir()` adds all the file(s) in a directory to a new archive.

Usage

```
archive_write_dir(
  archive,
  dir,
  format = NULL,
  filter = NULL,
  options = character(),
  ...,
  recursive = TRUE,
  full.names = FALSE
)

archive_write_files(
  archive,
  files,
  format = NULL,
  filter = NULL,
  options = character()
)
```

Arguments

`archive`	`character(1)` The archive filename or an `archive` object.
`dir`	`character(1)` The directory of files to add.
`format`	`character(1)` default: `NULL` The archive format, one of '7zip', 'cab', 'cpio', 'iso9660', 'lha', 'mtree', 'shar', 'rar', 'raw', 'tar', 'xar', 'zip', 'warc'.
`filter`	`character(1)` default: `NULL` The archive filter, one of 'none', 'gzip', 'bzip2', 'compress', 'lzma', 'xz', 'uuencode', 'lzip', 'lrzip', 'lzop', 'grzip', 'lz4', 'zstd'.
`options`	`character()` default: `character(0)` Options to pass to the filter or format. The list of available options are documented in options can have one of the following forms: • `option=value` The option/value pair will be provided to every module. Modules that do not accept an option with this name will ignore it. • `option` The option will be provided to every module with a value of "1". • `!option` The option will be provided to every module with a NULL value. • `module:option=value`, `module:option`, `module:!option` As above, but the corresponding option and value will be provided only to modules whose name matches module. See read options for available read options See write options for available write options
`...`	additional parameters passed to `base::dir`.

recursive	logical. Should the listing recurse into directories?
full.names	a logical value. If TRUE, the directory path is prepended to the file names to give a relative file path. If FALSE, the file names (rather than paths) are returned.
files	character() One or more files to add to the archive.

Value

An 'archive' object representing the new archive (invisibly).

An 'archive' object representing the new archive (invisibly).

Examples

```
if (archive:::libarchive_version() > "3.2.0") {
# write some files to a directory
d <- tempfile()
dir.create(d)
old <- setwd(d)

write.csv(iris, file.path(d, "iris.csv"))
write.csv(mtcars, file.path(d, "mtcars.csv"))
write.csv(airquality, file.path(d, "airquality.csv"))

# Add some to a new archive
a <- archive_write_files("data.tar.gz", c("iris.csv", "mtcars.csv"))
setwd(old)
a

# Add all files in a directory
a <- archive_write_dir("data.zip", d)
a

unlink("data.zip")
}
```

---

file_read *Construct a connections for (possibly compressed) files.*

Description

They are functionally equivalent to calling archive_read or archive_write using format = "raw", archive = file.

Usage

```
file_read(file, mode = "r", filter = NULL, options = character())

file_write(file, mode = "w", filter = NULL, options = character())
```

Arguments

file	character(1) || integer(1) The filename within the archive, specified either by filename or by position.
mode	character(1) A description of how to open the connection (if it should be opened initially). See section 'Modes' in base::connections() for possible values.
filter	character(1) default: NULL The archive filter, one of 'none', 'gzip', 'bzip2', 'compress', 'lzma', 'xz', 'uuencode', 'lzip', 'lrzip', 'lzop', 'grzip', 'lz4', 'zstd'.
options	character() default: character(0) Options to pass to the filter or format. The list of available options are documented in options can have one of the following forms:

- option=value The option/value pair will be provided to every module. Modules that do not accept an option with this name will ignore it.
- option The option will be provided to every module with a value of "1".
- !option The option will be provided to every module with a NULL value.
- module:option=value, module:option, module:!option As above, but the corresponding option and value will be provided only to modules whose name matches module. See read options for available read options See write options for available write options

Details

file_write() returns an writable output connection, file_read() returns a readable input connection.

Value

An 'archive_read' connection (for file_read()) or an 'archive_write' connection (for file_write()) to the file.

Examples

```
if (archive:::libarchive_version() > "3.2.0") {
# Write bzip2, base 64 encoded data and use high compression
write.csv(mtcars,
  file_write("mtcars.bz2",
    filter = c("uuencode", "bzip2"),
    options = "compression-level=9"
  )
)

# Read it back
read.csv(file_read("mtcars.bz2"), row.names = 1, nrows = 3)
unlink("mtcars.bz2")
}
```

Index