

# Package ‘biocmake’

December 4, 2025

**Version** 1.3.1

**Date** 2025-11-22

**Title** CMake for Bioconductor

**Description** Manages the installation of CMake for building Bioconductor packages.

This avoids the need for end-users to manually install CMake on their system.

No action is performed if a suitable version of CMake is already available.

**License** MIT + file LICENSE

**Encoding** UTF-8

**VignetteBuilder** knitr

**Imports** utils, tools, dir.expiry

**Suggests** knitr, rmarkdown, BiocStyle, testthat

**biocViews** Infrastructure

**URL** <https://github.com/LTLA/biocmake>

**BugReports** <https://github.com/LTLA/biocmake/issues>

**RoxygenNote** 7.3.2

**git\_url** <https://git.bioconductor.org/packages/biocmake>

**git\_branch** devel

**git\_last\_commit** db9e091

**git\_last\_commit\_date** 2025-11-21

**Repository** Bioconductor 3.23

**Date/Publication** 2025-12-04

**Author** Aaron Lun [cre, aut]

**Maintainer** Aaron Lun <[infinite.monkeys.with.keyboards@gmail.com](mailto:infinite.monkeys.with.keyboards@gmail.com)>

## Contents

|           |   |
|-----------|---|
| configure | 2 |
| defaults  | 3 |
| download  | 4 |
| find      | 5 |

configure

*Configure Cmake***Description**

Propagate R's configuration variables into the Cmake options, where possible.

**Usage**

```
configure(
  c.compiler = TRUE,
  c.flags = c.compiler,
  cxx.compiler = TRUE,
  cxx.flags = cxx.compiler,
  fortran.compiler = TRUE,
  fortran.flags = fortran.compiler,
  cpp.flags = c.compiler || cxx.compiler,
  pic.flags = TRUE,
  ld.flags = c("exe", "module", "shared"),
  make = TRUE,
  ar = TRUE,
  ranlib = TRUE,
  release.build = TRUE
)

formatArguments(options)
```

**Arguments**

|                               |                                                                                                                                                                                    |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>c.compiler</code>       | Logical scalar indicating whether to propagate R's choice of C compiler.                                                                                                           |
| <code>c.flags</code>          | Logical scalar indicating whether to propagate R's choice of C flags.                                                                                                              |
| <code>cxx.compiler</code>     | Logical scalar indicating whether to propagate R's choice of C++ compiler.                                                                                                         |
| <code>cxx.flags</code>        | Logical scalar indicating whether to propagate R's choice of C++ flags.                                                                                                            |
| <code>fortran.compiler</code> | Logical scalar indicating whether to propagate R's choice of Fortran compiler.                                                                                                     |
| <code>fortran.flags</code>    | Logical scalar indicating whether to propagate R's choice of Fortran flags.                                                                                                        |
| <code>cpp.flags</code>        | Logical scalar indicating whether to propagate R's choice of C/C++ preprocessing flags.                                                                                            |
| <code>pic.flags</code>        | Logical scalar indicating whether to propagate R's choice of each language's position-independent flags. This also sets the <code>CMAKE_POSITION_INDEPENDENT_CODE</code> variable. |
| <code>ld.flags</code>         | Logical scalar indicating whether to add R's choice of linker flags to the CMake variables for each target type.                                                                   |

|               |                                                                                                 |
|---------------|-------------------------------------------------------------------------------------------------|
| make          | Logical scalar indicating whether to propagate R's choice of make command.                      |
| ar            | Logical scalar indicating whether to propagate R's choice of command to make static libraries.  |
| ranlib        | Logical scalar indicating whether to propagate R's choice of command to index static libraries. |
| release.build | Logical scalar indicating whether to configure Cmake for a release build.                       |
| options       | Character vector of optional arguments from configure.                                          |

### Value

For `configure`, a named character vector containing the name and value of each option.

For `formatArguments`, a character vector with Cmake arguments on the command line. NA values are ignored, and values with spaces or empty strings are quoted.

### Author(s)

Aaron Lun

### Examples

```
options <- configure()
options
formatArguments(options)
```

---

defaults

*Defaults for **biocmake***

---

### Description

Defaults for **biocmake**

### Usage

```
defaultCommand()

defaultDownloadVersion()

defaultMinimumVersion()

defaultCacheDirectory()
```

## Details

The `BIOCMAKE_CMAKE_COMMAND` environment variable will override the default setting of `defaultCommand`.

The `BIOCMAKE_CMAKE_DOWNLOAD_VERSION` environment variable will override the default setting of `defaultDownloadVersion`.

The `BIOCMAKE_CMAKE_MINIMUM_VERSION` environment variable will override the default setting of `defaultMinimumVersion`.

The `BIOCMAKE_CMAKE_CACHE_DIRECTORY` environment variable will override the default setting of `defaultCacheDirectory`.

## Value

For `defaultCommand`, a string specifying the expected command-line invocation of an existing Cmake installation.

For `defaultDownloadVersion`, a string specifying the version of Cmake to download if no existing installation can be found.

For `defaultMinimumVersion`, a string specifying the minimum version of an existing Cmake installation.

For `defaultCacheDirectory`, a string containing the path to the cache directory for **biocmake**-managed Cmake installations.

## Author(s)

Aaron Lun

## Examples

```
defaultCommand()
defaultDownloadVersion()
defaultMinimumVersion()
defaultCacheDirectory()
```

---

download

*Download Cmake*

---

## Description

Download Cmake binaries for the current architecture. This uses **dir.expiry** to remove unused versions of the **biocmake**-managed Cmake.

## Usage

```
download(
  download.version = defaultDownloadVersion(),
  cache.dir = defaultCacheDirectory(),
  ignore.cache = FALSE
)
```

**Arguments**

|                  |                                                                                                                                        |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| download.version | String specifying the Cmake version to download.                                                                                       |
| cache.dir        | String specifying the location of the directory in which to cache Cmake installations.                                                 |
| ignore.cache     | Logical scalar specifying whether to ignore any existing cached version of Cmake, in which case the binaries will be downloaded again. |

**Value**

String containing the path to the Cmake executable.

**Author(s)**

Aaron Lun

**Examples**

```
download()
```

---

find

*Find Cmake*


---

**Description**

Find an existing Cmake installation or, if none can be found, install a **biocmake**-managed Cmake instance.

**Usage**

```
find(
  command = defaultCommand(),
  minimum.version = defaultMinimumVersion(),
  can.download = TRUE,
  forget = FALSE,
  ...
)
```

**Arguments**

|                 |                                                                                                          |
|-----------------|----------------------------------------------------------------------------------------------------------|
| command         | String containing the command to check for an existing installation.                                     |
| minimum.version | String specifying the minimum acceptable version of an existing installation.                            |
| can.download    | Logical scalar indicating whether to download Cmake if no acceptable existing installation can be found. |
| forget          | Logical scalar indicating whether to forget the results of the last call.                                |
| ...             | Further arguments to pass to <a href="#">download</a> .                                                  |

**Details**

If the `BIOCMAKE_FIND_OVERRIDE` environment variable is set to a command or path to a Cmake executable, it is returned directly and all other options are ignored.

On Windows, it is strongly recommended to download Rtools (see <https://cran.r-project.org/bin/windows/Rtools/rtools44/rtools.html>). This provides a pre-configured Cmake that is guaranteed to work.

By default, `find` will remember the result of its last call in the current R session, to avoid re-checking the versions, cache, etc. This can be disabled by setting `forget=TRUE` to force a re-check, e.g., to detect a new version of Cmake that was installed while the R session is active.

**Value**

String containing the command to use to run Cmake.

**Author(s)**

Aaron Lun

**Examples**

```
cmd <- find()
system2(cmd, "--version")
```

# Index

`configure`, [2](#)

`defaultCacheDirectory (defaults)`, [3](#)

`defaultCommand (defaults)`, [3](#)

`defaultDownloadVersion (defaults)`, [3](#)

`defaultMinimumVersion (defaults)`, [3](#)

`defaults`, [3](#)

`download`, [4](#), [5](#)

`find`, [5](#)

`formatArguments (configure)`, [2](#)