Package 'BiocBuildReporter'

November 5, 2026

Type Package

Title Functions to process a bioconductor build report database

Version 1.0.1

Description This package reads remote parquet files that have processed Bioconductor build report logs. Users may query the tables directly for specific information or use predefined helper functions for common queries. The logs processed are from https://bioconductor.org/checkResults/. In the future we will extend this package out to include processing of r-universe logs.

Depends R (>= 4.6.0)

Imports arrow, dplyr, BiocFileCache

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

Suggests BiocStyle, testthat (>= 3.0.0), knitr, rmarkdown, ggplot2, tidyr, stringr

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/lshep/BiocBuildReporter.git

BugReports https://github.com/lshep/BiocBuildReporter/issues

biocViews Software, Infrastructure

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

git_branch RELEASE_3_23

git_last_commit 1536fa2

git_last_commit_date 2026-05-20

Repository Bioconductor 3.23

Date/Publication 2026-11-05

Author Sean Davis [aut], Lori Shepherd [aut, cre] (ORCID: <https://orcid.org/0000-0002-3242-0582>)

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

Contents

bbs_cache . . . 2
BiocBuildReporter . . . 2
get_all_bbs_tables . . . 3
get_bbs_table . . . 4
get_build_report . . . 6
get_failing_packages . . . 7
get_latest_branches . . . 8
get_package_build_results . . . 8
get_package_release_info . . . 9
list_all_branches . . . 10
list_all_builders . . . 11
package_error_count . . . 12
package_failures_over_time . . . 13

Index **15**

`bbs_cache` *Custom Environment to store downloaded parquet tibbles.*

Description

A custom environment to store remotely read parquet files of Bioconductor build reports. Once one of the three valid parquet files are read in, the table is cached in this environment for quick easy reference either to use ad hocly by the user or to be used within pre-defined queries provided by this package.

Usage

```
.bbs_cache
```

Format

An environment

`BiocBuildReporter` *BiocBuildReporter: Functions to process a bioconductor build report database*

Description

The BiocBuildReporter package provides access to years of Bioconductor build system data, representing a comprehensive record of package builds across: - **Thousands of packages** in the Bioconductor ecosystem - **Multiple R versions** spanning several years of development - **Multiple platforms** including Linux, macOS, and Windows - **Different build stages** (install, build, check)

Details

Please see vignette("BiocBuildReporter") for details on usage and pre-built query functions.

Author(s)

Maintainer: Lori Shepherd <lori.shepherd@roswellpark.org> (ORCID)

Authors:

- Sean Davis <seandavi@gmail.com>

See Also

Useful links:

- https://github.com/lshep/BiocBuildReporter.git
- Report bugs at https://github.com/lshep/BiocBuildReporter/issues

---

`get_all_bbs_tables` *Retrieve all available Bioconductor build system parquet files*

Description

Retrieve all available parquet tables at once.

Usage

```
get_all_bbs_tables(
  assign_to_global = FALSE,
  useLocal = TRUE,
  updateLocal = FALSE
)
```

Arguments

`assign_to_global`	logical(1) indicating if downloaded tables should be assigned to the current global environment for access. Default is FALSE and only assigned to custom environment through call to get_bbs_table.
`useLocal`	A logical indicating if files should be read/downloaded from a local file or read remotely.
`updateLocal`	A logical indicating if using locally cached files, if the files should be re-downloaded/updated. Theorically build reports are updated daily so the local data may be out of date if not updated.

Details

This function is a quick wrapper around get_bbs_table to download all available parquet files as tibbles. By default no value is returned and the tibbles are stored to the custom environment and can be easily accessed through a call to get_bbs_table. If useLocal and updateLocal are TRUE, this also becomes an easy helper to update/redownloaded all files to the locally downloaded location.

Value

invisible. tables are cached to custom environment unless assign_to_global is TRUE.

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
get_all_bbs_tables()

# Now that all are downloaded. Retrieve cached build_summary tibble
build_summary <- get_bbs_table("build_summary")

library(stringr)
library(dplyr)
# find the times the package BiocFileCache failed on linux nebbiolo builders

build_summary |>
  filter(
    package == "BiocFileCache",
    status %in% c("TIMEOUT", "ERROR"),
    str_starts(node, "nebbiolo")
  )
```

---

get_bbs_table *Retrieve a remotely read parquet file of Bioconductor build data*

Description

Bioconductor Build results are at times cumbersome to work with and interrogate. Parquet files of condensed reports are provided remotely. These functions read the remote parquet files using the arrow package and return a tibble. Once retrieved the table is stored in a custom environment for easy quick reference either to use ad hocly or within pre-defined queries provided by this package. If useLocal is chosen, the remote parquet files are downloaded locally and cached using BiocFileCache. If already downloaded, the local file is utilized unless updateLocal is also TRUE in which case the file will be redownloaded to the local location before reading into the session. If useLocal is FALSE, the files will be read remotely.

Usage

```
get_bbs_table(
  tblname = c("build_summary", "info", "propagation_status"),
  useLocal = TRUE,
  updateLocal = FALSE
)
```

Arguments

tblname	A valid parquet file name. Currently available: build_summary, info, or propagation_status.
useLocal	A logical indicating if files should be read/downloaded locally or read remotely.
updateLocal	A logical indicating if using locally cached files, if the files should be re-downloaded/updated. Theorically build reports are updated daily so the local data may be out of date if not updated.

Details

The get_bbs_table returns a tibble but is also saved in a custom environment for quick easy subsequent reference. There are three tables available and can be specified in get_bbs_table: build_summary, info, or propagation_status. build_summary returns a tibble with the results of each stage of the Bioconductor build process. info returns a tibble with information regarding the package maintainer and git commit status. propagation_status returns a tibble with data on if a package propagated to the community.

Value

returns a tibble.

Author(s)

Lori Shepherd

See Also

get_all_bbs_tables

Examples

```
info <- get_bbs_table("info")
info

library(dplyr)
# all the package Lori Shepherd maintains
info |> filter(Maintainer == "Lori Shepherd") |> distinct(Package)
```

get_build_report *Get a report for any given day from the Bioconductor Build System*

Description

Get a report for any given day from the Bioconductor Build System, optionally specifying a git branch or build machine name.

Usage

```
get_build_report(build_date = Sys.Date(), branch = NULL, builder = NULL)
```

Arguments

build_date	Date of report to retrieve (eg. '2025-12-19')
branch	a bioconductor branch (eg. 'devel', 'RELEASE_3_22')
builder	name of a valid Bioconductor build machine (ie 'nebbiolo1')

Details

Utilizes data from the build_summary and info parquet files to retrieve a reconstructed Bioconductor Build Report for any given day. Optionally a given build machine name or Bioconductor branch may be specified for further filtering. By default it will use today's date and retrieve both devel and current release for all Bioconductor build machines. NOTE: Traditionally Bioconductor splits builds into software, data-experiment, annotation, workflow, books; this distinction is not available in these functions and packages regardless of 'type' will be included. The returned tibble includes: package name, build node, build stage, package version number, status of stage, when the run started and ended, the command the builder ran, branch of bioconductor and relevant git commit information.

Value

tibble representing a Bioconductor build report for a given day

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
get_build_report("2025-12-29", branch="RELEASE_3_22", builder="nebbiolo2")
```

`get_failing_packages` *Get list of failing packages for the most recent build*

Description

Get a list of failing (ERROR/TIMEOUT) packages for any given branch and builder

Usage

```
get_failing_packages(branch = NULL, builder = NULL)
```

Arguments

`branch`	a bioconductor branch (eg. 'devel', 'RELEASE_3_22')
`builder`	name of a valid Bioconductor build machine (ie 'nebbiolo1')

Details

Utilizes data from the build_summary and info parquet files to list all packages failing (ERROR/TIMEOUT) in any stage of the process. Optionally a Bioconductor git branch can be specified and by default will do both the current devel and release. The report can be further filtered by optionally specifying a build machine name. The returned tibble includes: git branch, package name, package version number, build machine node, collapsed column of stages and statuses.

Value

a tibble of failing packages

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
get_failing_packages("RELEASE_3_22")
get_failing_packages("RELEASE_3_22", "nebbiolo2")
```

get_latest_branches *Get currently active Bioconductor git branches*

Description

Get currently active Bioconductor git branches

Usage

```
get_latest_branches(infoTbl = NULL)
```

Arguments

infoTbl	downloaded paraquet info table retrieved from get_bbs_table('info') that includes git_branch information.

Details

Get currently active Bioconductor git branches

Value

character(2) 'devel' and RELEASE_X_Y value eqivalent to current release.

See Also

get_bbs_table

Examples

```
get_latest_branches()
```

get_package_build_results *Get the most recent build result data for a given package on a given branch*

Description

Get the build status information for a given package on a specified branch.

Usage

```
get_package_build_results(packagename, branch = "devel")
```

Arguments

packagename	a single valid Bioconductor package name
branch	a bioconductor branch (eg. 'devel', 'RELEASE_3_22')

Details

Utilizes data from the info and build_summary parquet files to retrieve the latest build results for a specified package on a given branch. If no branch is given it defaults to use 'devel'. The data returned includes the node (builder machine name), stage (install, build, check), package version number, status of stage, date stage completed, and git information for the build (git branch, git commit hash, git commit date).

Value

tibble of build status information or NULL if package or branch is not found

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
get_package_build_results("BiocFileCache")
get_package_build_results("BiocFileCache", branch="RELEASE_3_22")
```

---

get_package_release_info

Get git information for a given package across all Bioconductor releases

---

Description

Get the package version and git commit data for all releases in Bioconductor.

Usage

```
get_package_release_info(packagename)
```

Arguments

packagename	a single valid Bioconductor package name

Details

Utilizes data from the info parquet file to retrieve the equivalent git information for every available Bioconductor release of a package. This include the git commit hash and last commit date along with the package version.

Value

tibble of release information or NULL if package is not found

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
get_package_release_info("BiocFileCache")
```

---

list_all_branches *Get list of all Bioconductor git branches*

Description

Get list of allBioconductor git branches

Usage

```
list_all_branches(infoTbl = NULL)
```

Arguments

infoTbl	downloaded paraquet info table retrieved from get_bbs_table('info') that includes git_branch information.

Details

Get list of all Bioconductor git branches

Value

character() vector listing all options for "branch"

See Also

get_bbs_table

Examples

```
list_all_branches()
```

`list_all_builders` *Get list of all Bioconductor builders*

Description

Get list of allBioconductor builders

Usage

```
list_all_builders(summaryTbl = NULL)
```

Arguments

`summaryTbl`	downloaded paraquet build_summary table retrieved from get_bbs_table('build_summary') that includes builder (node) information.

Details

Get list of all Bioconductor builders

Value

character() vector listing all options for "builder"

See Also

get_bbs_table

Examples

```
list_all_builders()
```

`package_error_count` *Get a count of how many time a package failed on the Bioconductor Build System*

Description

Get a count of how many time a package failed (ERROR/TIMEOUT) on the Bioconductor Build System; optionally can specify a specific builder or branch to filter results.

Usage

```
package_error_count(packagename, builder = NULL, branch = NULL)
```

Arguments

`packagename`	a single valid Bioconductor package name
`builder`	name of a valid Bioconductor build machine (ie 'nebbiolo1')
`branch`	a bioconductor branch (eg. 'devel', 'RELEASE_3_22')

Details

Utilizes data from the build_summary and info parquet files to retrieve a count of errors (TIMEOUT/ERROR) for a specified package. Optionally a given builder machine name or Bioconductor branch may be specified to filter request. If no builder is given it will return all. If no branch is give it will also return all. The data returned includes the node (builder machine name), package version number, stage (install, build, check), total number of times that stage/builder tried to run, total number of times the package failed during that stage, and the given Bioconductor git branch. NOTE: since 'devel' is on-going. There will be multiple 'devel' results across multiple years not just the current devel. See example to filtering to see most recent. NOTE: If builder or branch is not found, function will continue and return unfiltered results.

Value

tibble of data including counts of runs and errors or NULL if package is not found

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
package_error_count("BiocFileCache")
package_error_count("BiocFileCache", branch="RELEASE_3_22")
package_error_count("BiocFileCache", builder = "nebbiolo2", branch="RELEASE_3_22")

library(dplyr)
## devel will have more than just current devel
devErrors <- package_error_count("BiocFileCache", branch="devel")
## can use package version number to filter to current devel
devErrors |> filter(version == max(version))
```

`package_failures_over_time` *Get rough timeline overview of package failing events*

Description

Get rough timeline overview of a package failing events on a specific builder

Usage

```
package_failures_over_time(packagename, builder, failure_cluster_hours = 72)
```

Arguments

`packagename`	a single valid Bioconductor package name
`builder`	name of a valid Bioconductor build machine (ie 'nebbiolo1')
`failure_cluster_hours`	numeric indicating the number of hours to treat failures as a single event

Details

This function attempts to give an overview of how long a package has been failing on a given builder. The time to treat failures as a single event is distinguished by consecutive days and by the failure_cluster_hours argument. Currently there is a different cadence of builds for release compared to devel so this allows the flexibility to define the timeline. Let's use the following example for interpretation of the result table: "' version episode first_failure last_failure n_failures 2 3.1.0 3 2025-12-15 02:57:57 2025-12-16 21:29:20 3 3 3.1.0 2 2025-12-13 02:32:33 2025-12-13 02:32:33 1 4 3.1.0 1 2025-12-08 21:30:14 2025-12-11 21:27:29 4 stages statuses 2 buildsrc, checksrc ERROR 3 checksrc ERROR 4 buildsrc ERROR "' The package version being built on the build system for this package is 3.1.0. The episode refers to the number of failing episodic events. If we look at the episode 1, the package started failing on December 8 and continued to fail until December 11. During that time period there were 4 number of failures and they occurred during the build stage. The time between the last failure of the first episode and the second episode is from December 11 to December 13; we can infer then that the package built cleanly on December 12. During episode 2 the package failed solely on December 13 but during the check stage. Episode

3 began on Dec 15 and lasted until December 16 and had 3 failures but they occurred in either the build or check stage.

This hopefully can help determine if package failures are intermittent and how long a package is failing on a given builder.

Value

tibble of failure event information or NULL if package or builder is not found

Author(s)

Lori Shepherd

See Also

get_bbs_table

Examples

```
package_failures_over_time("BiocFileCache", "nebbiolo1", 24)
```

Index

∗ **datasets**
bbs_cache, 2
.bbs_cache *(*bbs_cache*)*, 2

bbs_cache, 2
BiocBuildReporter, 2
BiocBuildReporter-package
*(*BiocBuildReporter*)*, 2

get_all_bbs_tables, 3
get_bbs_table, 4
get_build_report, 6
get_failing_packages, 7
get_latest_branches, 8
get_package_build_results, 8
get_package_release_info, 9

list_all_branches, 10
list_all_builders, 11

package_error_count, 12
package_failure_over_time
*(*package_failures_over_time*)*, 13
package_failures_over_time, 13