

Package ‘beastier’

August 7, 2020

Type Package

Title Call 'BEAST2'

Version 2.1.3

Maintainer Richèl J.C. Bilderbeek <richel@richelbilderbeek.nl>

Description 'BEAST2' (<<https://www.beast2.org>>) is a widely used Bayesian phylogenetic tool, that uses DNA/RNA/protein data and many model priors to create a posterior of jointly estimated phylogenies and parameters.
'BEAST2' is a command-line tool.
This package provides a way to call 'BEAST2' from an 'R' function call.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports ape, assertive, beautiful (>= 2.3.5), phangorn, pryr, rappdirs, remotes, rJava, stringr, xml2

Suggests hunspell, knitr, rmarkdown, spelling, testit, testthat (>= 2.1.0), tracerer

URL <https://docs.ropensci.org/beastier> (website)
<https://github.com/ropensci/beastier>

BugReports <https://github.com/ropensci/beastier>

Language en-US

VignetteBuilder knitr

SystemRequirements BEAST2 (<https://www.beast2.org/>)

NeedsCompilation no

Author Richèl J.C. Bilderbeek [aut, cre]
(<<https://orcid.org/0000-0003-1107-7049>>),
Joëlle Barido-Sottani [rev] (Joëlle reviewed the package for rOpenSci,
see <https://github.com/ropensci/onboarding/issues/209>),

David Winter [rev] (David reviewed the package for rOpenSci, see <https://github.com/ropensci/onboarding/issues/209>),
 Thijs Janzen [ctb]

Repository CRAN

Date/Publication 2020-08-07 09:40:06 UTC

R topics documented:

are_beast2_input_lines	4
are_beast2_input_lines_deep	5
are_beast2_input_lines_fast	6
are_identical_alignments	7
beast2_internal_filenames_to_table	7
beast2_options_to_table	8
beastier	8
beastier_report	9
check_beast2	9
check_beast2_internal_filenames	10
check_beast2_internal_filenames_data_types	11
check_beast2_internal_filenames_filenames_differ	12
check_beast2_internal_filenames_names	12
check_beast2_options	13
check_beast2_optionses	14
check_beast2_options_data_types	15
check_beast2_options_do_not_overwrite_existing_files	15
check_beast2_options_filenames_differ	16
check_beast2_options_names	17
check_beast2_path	17
check_can_create_file	18
check_input_filename	19
check_n_threads	19
check_os	20
check_rng_seed	21
create_beast2_internal_filenames	22
create_beast2_options	23
create_beast2_run_cmd	25
create_beast2_validate_cmd	26
create_beast2_validate_cmd_bin	27
create_beast2_validate_cmd_jar	28
create_beast2_version_cmd	29
create_beast2_version_cmd_bin	30
create_beast2_version_cmd_jar	30
create_temp_input_filename	31
create_temp_state_filename	32
default_params_doc	32
do_minimal_run	36
get_alignment_ids_from_xml_filename	36

get_beast2_example_filename	37
get_beast2_example_filenames	38
get_beast2_main_class_name	38
get_beast2_options_filenames	39
get_beast2_version	39
get_beastier_path	40
get_beastier_paths	41
get_default_beast2_bin_path	42
get_default_beast2_download_url	43
get_default_beast2_download_url_linux	44
get_default_beast2_download_url_win	44
get_default_beast2_folder	45
get_default_beast2_jar_path	45
get_default_beast2_path	46
get_default_beast2_version	47
get_default_java_path	48
get_duplicate_param_ids	48
get_java_version	49
get_trees_filenames	50
gives_beast2_warning	51
has_unique_ids	52
install_beast2	53
is_alignment	54
is_beast2_input_file	54
is_beast2_installed	56
is_bin_path	56
is_jar_path	57
is_on_appveyor	58
is_on_ci	58
is_on_travis	59
print_beast2_internal_filenames	60
print_beast2_options	60
remove_file_if_present	61
rename_beast2_options_filenames	61
run_beast2	62
run_beast2_from_options	64
save_lines	64
save_nexus_as_fasta	65
uninstall_beast2	66
update_beastier	67
upgrade_beast2	67

are_beast2_input_lines

Would these lines of text, when written to a file, result in a valid BEAST2 input file?

Description

Would these lines of text, when written to a file, result in a valid BEAST2 input file?

Usage

```
are_beast2_input_lines(  
    lines,  
    verbose = FALSE,  
    method = ifelse(is_on_ci(), "deep", "fast"),  
    beast2_path = get_default_beast2_path()  
)
```

Arguments

lines	lines of text
verbose	if TRUE, additional information is displayed, that is potentially useful in debugging
method	the method to check. Can be 'deep' or 'fast'. The 'deep' method uses BEAST2 to validate the complete file. The 'fast' method uses some superficial tests (for example: if all IDs are unique)
beast2_path	name of either a BEAST2 binary file (usually simply beast) or a BEAST2 jar file (usually has a .jar extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path

Value

TRUE if the text is valid, FALSE if not

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [is_beast2_input_file](#) to check a file

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {
  beast2_filename <- get_beastier_path("anthus_2_4.xml")
  text <- readLines(beast2_filename)
  expect_true(are_beast2_input_lines(text))
}
```

are_beast2_input_lines_deep

Would these lines of text, when written to a file, result in a valid BEAST2 input file?

Description

Would these lines of text, when written to a file, result in a valid BEAST2 input file?

Usage

```
are_beast2_input_lines_deep(
  lines,
  verbose = FALSE,
  beast2_path = get_default_beast2_path()
)
```

Arguments

lines	lines of text
verbose	if TRUE, additional information is displayed, that is potentially useful in debugging
beast2_path	name of either a BEAST2 binary file (usually simply <code>beast</code>) or a BEAST2 jar file (usually has a <code>.jar</code> extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path

Value

TRUE if the text is valid, FALSE if not

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [is_beast2_input_file](#) to check a file

Examples

```
if (is_beast2_installed() && is_on_ci()) {  
  beast2_filename <- get_beastier_path("anthus_2_4.xml")  
  text <- readLines(beast2_filename)  
  testit::assert(are_beast2_input_lines_deep(text))  
}
```

are_beast2_input_lines_fast

Would these lines of text, when written to a file, result in a valid BEAST2 input file?

Description

Would these lines of text, when written to a file, result in a valid BEAST2 input file?

Usage

```
are_beast2_input_lines_fast(lines)
```

Arguments

lines lines of text

Value

TRUE if the text is valid, FALSE if not

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [is_beast2_input_file](#) to check a file

Examples

```
library(testthat)  
  
beast2_filename <- get_beastier_path("anthus_2_4.xml")  
text <- readLines(beast2_filename)  
expect_true(are_beast2_input_lines_fast(text))
```

are_identical_alignments

Determines if the two alignments are equal

Description

Determines if the two alignments are equal

Usage

are_identical_alignments(p, q)

Arguments

p	the first alignment
q	the second alignment

Value

TRUE or FALSE

Author(s)

Richèl J.C. Bilderbeek

beast2_internal_filenames_to_table

Convert a beast2_internal_filenames to a table

Description

Convert a beast2_internal_filenames to a table

Usage

beast2_internal_filenames_to_table(beast2_internal_filenames)

Arguments

beast2_internal_filenames
a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

beast2_options_to_table

Convert a beast2_options to a table

Description

Convert a `beast2_options` to a table

Usage

```
beast2_options_to_table(beast2_options)
```

Arguments

`beast2_options` a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

beastier

beastier: A package to call BEAST2.

Description

`beastier` allows to call BEAST2, a popular Bayesian phylogenetics tool, using an R interface. 'beastier' closely follows the interface of BEAST2, including its default settings.

See Also

These are packages associated with `beastier`:

- The package `beautier` can create BEAST2 input files from R
- The package `tracerer` can parse BEAST2 output files from R
- The package `babette` combines the functionality of `beautier`, `beastier` and `tracerer` into a single workflow

Examples

```
library(testthat)

beast2_options <- create_beast2_options(
  input_filename = get_beastier_path("2_4.xml")
)

if (is_beast2_installed() && is_on_ci()) {
  expect_false(file.exists(beast2_options$output_state_filename))
}
```



```
output <- run_beast2_from_options(beast2_options)

expect_true(length(output) > 40)
expect_true(file.exists(beast2_options$output_state_filename))
}
```

beastier_report *Create a [beastier](#) report, to be used when reporting bugs*

Description

Create a [beastier](#) report, to be used when reporting bugs

Usage

```
beastier_report()
```

check_beast2 *Check if BEAST2 is installed properly.*

Description

Calls stop if BEAST2 is improperly installed

Usage

```
check_beast2(beast2_path = get_default_beast2_path())
```

Arguments

beast2_path name of either a BEAST2 binary file (usually simply `beast`) or a BEAST2 jar file (usually has a `.jar` extension). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed()) {
  expect_silent(check_beast2())
} else {
  expect_error(check_beast2())
}
```

check_beast2_internal_filenames

Check if the `beast2_internal_filenames` is a valid BEAST2 internal filenames object.

Description

Calls stop if the BEAST2 internal filenames object is invalid

Usage

```
check_beast2_internal_filenames(beast2_internal_filenames)
```

Arguments

`beast2_internal_filenames`
a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [create_beast2_internal_filenames](#) to create a valid BEAST2 internal filenames object

Examples

```
library(testthat)

if (beastier::is_beast2_installed()) {
  expect_silent(
    check_beast2_internal_filenames(
      create_beast2_internal_filenames(
        create_beast2_options(
          input_filename = get_beastier_path("2_4.xml")
        )
      )
    )
  )
}
```

```
    )
  )
)
}

# Must stop on nonsense
expect_error(check_beast2_internal_filenames("nonsense"))
expect_error(check_beast2_internal_filenames(NULL))
expect_error(check_beast2_internal_filenames(NA))
```

check_beast2_internal_filenames_data_types

Check if the `beast2_internal_filenames`, which is a list, has all elements of the right data types

Description

Calls stop if not.

Usage

```
check_beast2_internal_filenames_data_types(beast2_internal_filenames)
```

Arguments

beast2_internal_filenames
a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [check_beast2_internal_filenames](#) to check the entire `beast2_internal_filenames` object

check_beast2_internal_filenames_filenames_differ

Check if the filenames in beast2_internal_filenames differ

Description

Calls stop if not.

Usage

```
check_beast2_internal_filenames_filenames_differ(beast2_internal_filenames)
```

Arguments

beast2_internal_filenames

a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [check_beast2_internal_filenames](#) to check the entire beast2_internal_filenames object

check_beast2_internal_filenames_names

Check if the beast2_internal_filenames, which is a list, has all the elements needed.

Description

Calls stop if not.

Usage

```
check_beast2_internal_filenames_names(beast2_internal_filenames)
```

Arguments

beast2_internal_filenames

a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [check_beast2_internal_filenames](#) to check the entire `beast2_internal_filenames` object

`check_beast2_options` *Check if the `beast2_options` is a valid BEAST2 options object.*

Description

Calls `stop` if the BEAST2 option object is invalid

Usage

```
check_beast2_options(beast2_options)
```

Arguments

`beast2_options` a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [create_beast2_options](#) to create a valid BEAST2 options object

Examples

```
library(testthat)

expect_silent(check_beast2_options(create_beast2_options()))

# Must stop on nonsense
expect_error(check_beast2_options("nonsense"))
expect_error(check_beast2_options(NULL))
expect_error(check_beast2_options(NA))
```

check_beast2_optionses

Check if the beast2_options is a valid BEAST2 options object.

Description

Calls stop if the BEAST2 option object is invalid

Usage

```
check_beast2_optionses(beast2_optionses)
```

Arguments

beast2_optionses

list of one or more `beast2_options` structures, as can be created by [create_beast2_options](#).
Use of reduplicated plural to achieve difference with `beast2_options`

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [create_beast2_options](#) to create a valid BEAST2 options object

Examples

```
library(testthat)

expect_silent(check_beast2_optionses(list(create_beast2_options())))

# Must stop on nonsense
expect_error(check_beast2_optionses("nonsense"))
expect_error(check_beast2_optionses(NULL))
expect_error(check_beast2_optionses(NA))
```

`check_beast2_options_data_types`

Check if the `beast2_options`, which is a list, has all elements of the right data types

Description

Calls `stop` if not.

Usage

```
check_beast2_options_data_types(beast2_options)
```

Arguments

`beast2_options` a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [check_beast2_options](#) to check the entire `beast2_options` object

`check_beast2_options_do_not_overwrite_existing_files`

Check if the `beast2_options` will not overwrite existing files, when the 'overwrite' options is set to FALSE

Description

Will `stop` if a file is threatened to be overwritten

Usage

```
check_beast2_options_do_not_overwrite_existing_files(  
  beast2_options,  
  beast2_internal_filenames  
)
```

Arguments

beast2_options a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

beast2_internal_filenames
a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

Author(s)

Richèl J.C. Bilderbeek

check_beast2_options_filenames_differ

Check if the filenames in beast2_options differ

Description

Calls stop if not.

Usage

```
check_beast2_options_filenames_differ(beast2_options)
```

Arguments

beast2_options a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [check_beast2_options](#) to check the entire beast2_options object

check_beast2_options_names

Check if the beast2_options, which is a list, has all the elements needed.

Description

Calls stop if not.

Usage

```
check_beast2_options_names(beast2_options)
```

Arguments

beast2_options a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Value

nothing

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [check_beast2_options](#) to check the entire beast2_options object

check_beast2_path

Checks the BEAST2 .jar path. Will stop if there is a problem with the BEAST2 .jar path.

Description

Checks the BEAST2 .jar path. Will stop if there is a problem with the BEAST2 .jar path.

Usage

```
check_beast2_path(beast2_path)
```

Arguments

beast2_path name of either a BEAST2 binary file (usually simply beast) or a BEAST2 jar file (usually has a .jar extension). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Value

nothing. Will call `stop` if the BEAST2 .jar path has a problem

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed()) {
  beast2_path <- get_default_beast2_jar_path()
  expect_silent(check_beast2_path(beast2_path))
}
```

`check_can_create_file` *Check that a file can be created at a certain path.*

Description

Will `stop` if not. Will `stop` if the file already exists. Does so by creating an empty file at the path, and then deleting it.

Usage

```
check_can_create_file(filename, overwrite = TRUE)
```

Arguments

`filename` file that may or may not be created
`overwrite` if TRUE, if `filename` already exists, it will be deleted by this function

Author(s)

Richèl J.C. Bilderbeek

check_input_filename *Checks the input filename. Will stop if there is a problem with the input filename.*

Description

Checks the input filename. Will stop if there is a problem with the input filename.

Usage

```
check_input_filename(input_filename)
```

Arguments

input_filename the name of a BEAST2 input XML file. This file usually has an .xml extension. Use [create_temp_input_filename](#) to create a temporary filename with that extension.

Value

nothing. Will call [stop](#) if the input file is invalid

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_error(check_input_filename(tempfile()))
expect_silent(
  check_input_filename(
    get_beastier_path("beast2_example_output.log")
  )
)
```

check_n_threads *Check if the input is a valid number of threads.*

Description

Will [stop](#) if not.

Usage

```
check_n_threads(n_threads)
```

Arguments

n_threads the number of computational threads to use. Use [NA](#) to use the BEAST2 default of 1.

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_silent(check_n_threads(1))
expect_silent(check_n_threads(2))
expect_silent(check_n_threads(NA))

expect_error(check_n_threads(-1))
expect_error(check_n_threads("nonsense"))
expect_error(check_n_threads(c(1, 2)))
expect_error(check_n_threads(c()))
expect_error(check_n_threads(NULL))
```

check_os

Checks if the operating system is supported

Description

Checks if the operating system is supported

Usage

```
check_os(os)
```

Arguments

os name of the operating system, must be unix (Linux, Mac) or win (Windows)

Value

nothing. Will stop if the OS is unsupported

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_silent(check_os("mac"))
expect_silent(check_os("unix"))
expect_silent(check_os("win"))

expect_error(check_os("android"))
expect_error(check_os("n64"))
expect_error(check_os("nds"))
expect_error(check_os("nes"))
expect_error(check_os("snes"))

expect_error(check_os(NA))
expect_error(check_os(NULL))
expect_error(check_os(""))
expect_error(check_os(c()))
```

check_rng_seed	<i>Check if the input is a valid RNG seed.</i>
----------------	--

Description

Will [stop](#) if not.

Usage

```
check_rng_seed(rng_seed)
```

Arguments

rng_seed	the random number generator seed of the BEAST2 run. Must be a non-zero positive integer value or NA . If rng_seed is NA , BEAST2 will pick a random seed
----------	--

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_silent(check_rng_seed(1))
expect_silent(check_rng_seed(2))
expect_silent(check_rng_seed(NA))

expect_error(check_rng_seed(0))
expect_error(check_rng_seed(-1))
```

```
expect_error(check_rng_seed("nonsense"))
expect_error(check_rng_seed(c(1, 2)))
expect_error(check_rng_seed(c()))
expect_error(check_rng_seed(NULL))
```

create_beast2_internal_filenames

Create a list with the internally used BEAST2 filenames

Description

Create a list with the internally used BEAST2 filenames

Usage

```
create_beast2_internal_filenames(beast2_options)
```

Arguments

`beast2_options` a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Value

a list with the internally used BEAST2 filenames

Examples

```
library(testthat)

beast2_options <- create_beast2_options(
  input_filename = get_beastier_path("2_4.xml")
)
if (is_beast2_installed()) {
  beast2_internal_filenames <- create_beast2_internal_filenames(
    beast2_options
  )
  bif_names <- names(beast2_internal_filenames)
  expect_true("input_filename_full" %in% bif_names)
  expect_true("output_state_filename_full" %in% bif_names)
}
```

create_beast2_options *Function to create a set of BEAST2 options.*

Description

These BEAST2 options are the R equivalent of the command-line options.

Usage

```
create_beast2_options(  
  input_filename = create_temp_input_filename(),  
  output_state_filename = create_temp_state_filename(),  
  rng_seed = NA,  
  n_threads = NA,  
  use_beagle = FALSE,  
  overwrite = TRUE,  
  beast2_path = get_default_beast2_path(),  
  verbose = FALSE,  
  output_log_filename = "deprecated",  
  output_trees_filenames = "deprecated",  
  beast2_working_dir = "deprecated"  
)
```

Arguments

input_filename	the name of a BEAST2 input XML file. This file usually has an .xml extension. Use create_temp_input_filename to create a temporary filename with that extension.
output_state_filename	name of the .xml .state file to create. Use create_temp_state_filename to create a temporary filename with that extension.
rng_seed	the random number generator seed of the BEAST2 run. Must be a non-zero positive integer value or NA . If rng_seed is NA , BEAST2 will pick a random seed
n_threads	the number of computational threads to use. Use NA to use the BEAST2 default of 1.
use_beagle	use BEAGLE if present
overwrite	if TRUE: overwrite the .log and .trees files if one of these exists. If FALSE, BEAST2 will not be started if <ul style="list-style-type: none">• the .log file exists• the .trees files exist• the .log file created by BEAST2 exists• the .trees files created by BEAST2 exist

beast2_path	name of either a BEAST2 binary file (usually simply <code>beast</code>) or a BEAST2 jar file (usually has a <code>.jar</code> extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path
verbose	if TRUE, additional information is displayed, that is potentially useful in debugging
output_log_filename	name of the <code>.log</code> file to create
output_trees_filenames	one or more names for <code>.trees</code> file to create. There will be one <code>.trees</code> file created per alignment in the input file. The number of alignments must equal the number of <code>.trees</code> filenames, else an error is thrown. Alignments are sorted alphabetically by their IDs
beast2_working_dir	a folder where BEAST2 can work in isolation. For each BEAST2 run, a new subfolder is created in that folder. Within this folder, BEAST2 is allowed to create all of its output files, without the risk of overwriting existing ones, allowing BEAST2 to run in multiple parallel processes.

Value

a BEAST2 options structure

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

beast2_options <- create_beast2_options()

expect_true("input_filename" %in% names(beast2_options))
expect_true("output_state_filename" %in% names(beast2_options))
expect_true("rng_seed" %in% names(beast2_options))
expect_true("n_threads" %in% names(beast2_options))
expect_true("use_beagle" %in% names(beast2_options))
expect_true("overwrite" %in% names(beast2_options))
expect_true("beast2_path" %in% names(beast2_options))
expect_true("verbose" %in% names(beast2_options))

expect_silent(check_beast2_options(beast2_options))
```

create_beast2_run_cmd *Creates the terminal command to run BEAST2*

Description

Creates the terminal command to run BEAST2

Usage

```
create_beast2_run_cmd(
  input_filename,
  output_state_filename,
  rng_seed = NA,
  n_threads = NA,
  use_beagle = FALSE,
  overwrite = FALSE,
  beast2_path = get_default_beast2_path()
)
```

Arguments

input_filename	the name of a BEAST2 input XML file. This file usually has an <code>.xml</code> extension. Use create_temp_input_filename to create a temporary filename with that extension.
output_state_filename	name of the BEAST2 output file that stores the state (usually has a <code>.xml.state</code> extension)
rng_seed	the random number generator seed of the BEAST2 run. Must be a non-zero positive integer value or <code>NA</code> . If <code>rng_seed</code> is <code>NA</code> , BEAST2 will pick a random seed
n_threads	the number of computational threads to use. Use <code>NA</code> to use the BEAST2 default of 1.
use_beagle	use BEAGLE if present
overwrite	if TRUE: overwrite the <code>.log</code> and <code>.trees</code> files if one of these exists. If FALSE, BEAST2 will not be started if <ul style="list-style-type: none"> • the <code>.log</code> file exists • the <code>.trees</code> files exist • the <code>.log</code> file created by BEAST2 exists • the <code>.trees</code> files created by BEAST2 exist
beast2_path	name of either a BEAST2 binary file (usually simply <code>beast</code>) or a BEAST2 jar file (usually has a <code>.jar</code> extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path

Value

a character vector with the command and arguments to call BEAST2

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed()) {
  cmds <- create_beast2_run_cmd(
    input_filename = "input.xml",
    output_state_filename = "output.xml.state",
    beast2_path = get_default_beast2_jar_path()
  )
  testit::assert(cmds[2] == "-cp")
}
```

create_beast2_validate_cmd

Creates the terminal command to validate a BEAST2 input file

Description

Creates the terminal command to validate a BEAST2 input file

Usage

```
create_beast2_validate_cmd(
  input_filename,
  beast2_path = get_default_beast2_path()
)
```

Arguments

`input_filename` the name of a BEAST2 input XML file. This file usually has an `.xml` extension. Use [create_temp_input_filename](#) to create a temporary filename with that extension.

`beast2_path` name of either a BEAST2 binary file (usually simply `beast`) or a BEAST2 jar file (usually has a `.jar` extension). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Value

a character vector, of which the first element is the command (`java`, in this case), and the others are arguments (`-jar`, in this case, followed by more arguments).

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed() && is_on_ci()) {  
  cmds <- create_beast2_validate_cmd(  
    input_filename = "input.xml"  
  )  
  testit::assert(cmds[2] == "-cp")  
}
```

create_beast2_validate_cmd_bin

Creates the terminal command to validate a BEAST2 input file using a call to the launcher.jar file

Description

Creates the terminal command to validate a BEAST2 input file using a call to the launcher.jar file

Usage

```
create_beast2_validate_cmd_bin(  
  input_filename,  
  beast2_bin_path = get_default_beast2_bin_path()  
)
```

Arguments

input_filename the name of a BEAST2 input XML file. This file usually has an .xml extension. Use [create_temp_input_filename](#) to create a temporary filename with that extension.

beast2_bin_path name of the BEAST2 binary file (usually simply beast). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path

Value

a character vector, of which the first element is the command (java, in this case), and the others are arguments (-jar, in this case, followed by more arguments).

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed() && is_on_ci()) {  
  cmds <- create_beast2_validate_cmd_bin(  
    input_filename = "input.xml"  
  )  
  testit::assert(length(cmds) == 3)  
  testit::assert(cmds[2] == "-validate")  
}
```

create_beast2_validate_cmd_jar

Creates the terminal command to validate a BEAST2 input file using a call to the launcher.jar file

Description

Creates the terminal command to validate a BEAST2 input file using a call to the launcher.jar file

Usage

```
create_beast2_validate_cmd_jar(  
  input_filename,  
  beast2_jar_path = get_default_beast2_jar_path()  
)
```

Arguments

input_filename the name of a BEAST2 input XML file. This file usually has an .xml extension. Use [create_temp_input_filename](#) to create a temporary filename with that extension.

beast2_jar_path name of the BEAST2 jar file (usually has a .jar extension). Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Value

a character vector, of which the first element is the command (java, in this case), and the others are arguments (-jar, in this case, followed by more arguments).

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed() && is_on_ci()) {
  cmds <- create_beast2_validate_cmd_jar(
    input_filename = "input.xml"
  )
  testit::assert(length(cmds) == 6)
  testit::assert(cmds[2] == "-cp")
}
```

create_beast2_version_cmd

Creates the terminal command to version a BEAST2 input file

Description

Creates the terminal command to version a BEAST2 input file

Usage

```
create_beast2_version_cmd(beast2_path = beastier::get_default_beast2_path())
```

Arguments

beast2_path name of either a BEAST2 binary file (usually simply `beast`) or a BEAST2 jar file (usually has a `.jar` extension). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Value

a character vector, of which the first element is the command (`java`, in this case), and the others are arguments (`-jar`, in this case, followed by more arguments).

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed() && is_on_ci()) {
  cmds <- create_beast2_version_cmd()
  testit::assert(cmds[2] == "-cp")
}
```

create_beast2_version_cmd_bin

Creates the terminal command to version a BEAST2 input file using a call to the launcher . jar file

Description

Creates the terminal command to version a BEAST2 input file using a call to the launcher . jar file

Usage

```
create_beast2_version_cmd_bin(beast2_bin_path = get_default_beast2_bin_path())
```

Arguments

beast2_bin_path

name of the BEAST2 binary file (usually simply beast). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path

Value

a character vector, of which the first element is the command (java, in this case), and the others are arguments (-jar, in this case, followed by more arguments).

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed() && is_on_ci()) {  
  cmds <- create_beast2_version_cmd_bin()  
  testit::assert(length(cmds) == 2)  
  testit::assert(cmds[2] == "-version")  
}
```

create_beast2_version_cmd_jar

Creates the terminal command to version a BEAST2 input file using a call to the launcher . jar file

Description

Creates the terminal command to version a BEAST2 input file using a call to the launcher . jar file

Usage

```
create_beast2_version_cmd_jar(beast2_jar_path = get_default_beast2_jar_path())
```

Arguments

beast2_jar_path

name of the BEAST2 jar file (usually has a .jar extension). Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Value

a character vector, of which the first element is the command (java, in this case), and the others are arguments (-jar, in this case, followed by more arguments).

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed()) {  
  cmds <- create_beast2_version_cmd_jar()  
  testit::assert(length(cmds) == 5)  
  testit::assert(cmds[2] == "-cp")  
}
```

create_temp_input_filename

Create a temporary filename for the BEAST2 XML filename

Description

Create a temporary filename for the BEAST2 XML filename

Usage

```
create_temp_input_filename()
```

```
create_temp_state_filename
```

Create a temporary file for the BEAST2 XML output file that stores its state.

Description

Create a temporary file for the BEAST2 XML output file that stores its state.

Usage

```
create_temp_state_filename()
```

```
default_params_doc
```

This function does nothing. It is intended to inherit its parameters' documentation.

Description

This function does nothing. It is intended to inherit its parameters' documentation.

Usage

```
default_params_doc(
  beast2_bin_path,
  beast2_folder,
  beast2_internal_filenames,
  beast2_jar_path,
  beast2_options,
  beast2_optionses,
  beast2_path,
  beast2_version,
  beast2_working_dir,
  clock_model,
  clock_models,
  crown_age,
  crown_ages,
  fasta_filename,
  fasta_filenames,
  fixed_crown_age,
  fixed_crown_ages,
  initial_phylogenies,
  input_filename,
  mcmc,
  misc_options,
```



```

n_taxa,
n_threads,
os,
output_filename,
output_log_filename,
output_state_filename,
output_trees_filenames,
overwrite,
rename_fun,
rng_seed,
sequence_length,
site_model,
site_models,
tree_prior,
tree_priors,
use_beagle,
verbose
)

```

Arguments

- beast2_bin_path** name of the BEAST2 binary file (usually simply `beast`). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path
- beast2_folder** the folder where the BEAST2 is installed. Note that this is not the folder where the BEAST2 executable is installed: the BEAST2 executable is in a subfolder. Use [get_default_beast2_folder](#) to get the default BEAST2 folder. Use [get_default_beast2_bin_path](#) to get the full path to the default BEAST2 executable.
- beast2_internal_filenames** a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)
- beast2_jar_path** name of the BEAST2 jar file (usually has a `.jar` extension). Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path
- beast2_options** a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)
- beast2_optionses** list of one or more `beast2_options` structures, as can be created by [create_beast2_options](#). Use of reduplicated plural to achieve difference with `beast2_options`
- beast2_path** name of either a BEAST2 binary file (usually simply `beast`) or a BEAST2 jar file (usually has a `.jar` extension). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path
- beast2_version** the version of BEAST2. By default, this is the version as returned by [get_default_beast2_version](#)
- beast2_working_dir** a folder where BEAST2 can work in isolation. For each BEAST2 run, a new subfolder is created in that folder. Within this folder, BEAST2 is allowed to cre-

	ate all of its output files, without the risk of overwriting existing ones, allowing BEAST2 to run in multiple parallel processes.
clock_model	a <code>beautier</code> clock model
clock_models	a list of one or more <code>beautier</code> clock models
crown_age	the crown age of the phylogeny
crown_ages	the crown ages of the phylogenies. Set to <code>NA</code> if the crown age needs to be estimated
fasta_filename	a FASTA filename.
fasta_filenames	One or more FASTA filenames.
fixed_crown_age	determines if the phylogeny's crown age is fixed. If <code>FALSE</code> , crown age is estimated by BEAST2. If <code>TRUE</code> , the crown age is fixed to the crown age of the initial phylogeny.
fixed_crown_ages	one or more booleans to determine if the phylogenies' crown ages are fixed. If <code>FALSE</code> , crown age is estimated by BEAST2. If <code>TRUE</code> , the crown age is fixed to the crown age of the initial phylogeny.
initial_phylogenies	one or more MCMC chain's initial phylogenies. Each one set to <code>NA</code> will result in BEAST2 using a random phylogeny. Else the phylogeny is assumed to be of class <code>ape::phylo</code> .
input_filename	the name of a BEAST2 input XML file. This file usually has an <code>.xml</code> extension. Use create_temp_input_filename to create a temporary filename with that extension.
mcmc	one <code>beautier</code> MCMC
misc_options	one <code>beautier</code> <code>misc_options</code> object
n_taxa	The number of taxa
n_threads	the number of computational threads to use. Use <code>NA</code> to use the BEAST2 default of 1.
os	name of the operating system, must be <code>unix</code> (Linux, Mac) or <code>win</code> (Windows)
output_filename	Name of the XML parameter file created by this function. BEAST2 uses this file as input.
output_log_filename	name of the <code>.log</code> file to create
output_state_filename	name of the <code>.xml</code> state file to create. Use create_temp_state_filename to create a temporary filename with that extension.
output_trees_filenames	one or more names for <code>.trees</code> file to create. There will be one <code>.trees</code> file created per alignment in the input file. The number of alignments must equal the number of <code>.trees</code> filenames, else an error is thrown. Alignments are sorted alphabetically by their IDs

overwrite	if TRUE: overwrite the .log and .trees files if one of these exists. If FALSE, BEAST2 will not be started if <ul style="list-style-type: none"> • the .log file exists • the .trees files exist • the .log file created by BEAST2 exists • the .trees files created by BEAST2 exist
rename_fun	a function to rename a filename, as can be checked by check_rename_fun . This function should have one argument, which will be a filename or <code>NA</code> . The function should return one filename (when passed one filename) or one <code>NA</code> (when passed one <code>NA</code>). Example rename functions are: <ul style="list-style-type: none"> • get_remove_dir_fun get a function that removes the directory paths from the filenames, in effect turning these into local files • get_replace_dir_fun get a function that replaces the directory paths from the filenames • get_remove_hex_fun get a function that removes the hex string from filenames. For example, <code>tracelog_82c1a522040.log</code> becomes <code>tracelog.log</code>
rng_seed	the random number generator seed of the BEAST2 run. Must be a non-zero positive integer value or <code>NA</code> . If <code>rng_seed</code> is <code>NA</code> , BEAST2 will pick a random seed
sequence_length	a DNA sequence length, in base pairs
site_model	a beautier site model
site_models	one or more beautier site models
tree_prior	a beautier tree prior
tree_priors	one or more beautier tree priors
use_beagle	use BEAGLE if present
verbose	if TRUE, additional information is displayed, that is potentially useful in debugging

Value

Nothing. This is an internal function that does nothing

Note

This is an internal function, so it should be marked with `@noRd`. This is not done, as this will disallow all functions to find the documentation parameters

Author(s)

Richèl J.C. Bilderbeek

`do_minimal_run` *Do a minimal BEAST2 run*

Description

To achieve this, [run_beast2_from_options](#) is called.

Usage

```
do_minimal_run()
```

Value

The text sent to STDOUT and STDERR. It will create the files with name `output_state_filename`

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {
  expect_silent(do_minimal_run())
}
```

`get_alignment_ids_from_xml_filename`

Get the alignment ID from a file with one alignment

Description

Get the alignment ID from a file with one alignment

Usage

```
get_alignment_ids_from_xml_filename(xml_filename)
```

Arguments

`xml_filename` name of a BEAST2 XML input filename

Value

one or more alignment IDs

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_equal(
  get_alignment_ids_from_xml_filename(get_beastier_path("2_4.xml")),
  "test_output_0"
)
expect_equal(
  get_alignment_ids_from_xml_filename(
    get_beastier_path("anthus_15_15.xml")
  ),
  c("anthus_aco", "anthus_nd2")
)
```

get_beast2_example_filename

Get the full path of a BEAST2 example file

Description

Will [stop](#) if the filename is not a BEAST2 example file

Usage

```
get_beast2_example_filename(
  filename,
  beast2_folder = get_default_beast2_folder()
)
```

Arguments

filename	name of the BEAST2 example file. This should exclude the full path; this function exists to add that full path
beast2_folder	the folder where the BEAST2 is installed. Note that this is not the folder where the BEAST2 executable is installed: the BEAST2 executable is in a sub-folder. Use get_default_beast2_folder to get the default BEAST2 folder. Use get_default_beast2_bin_path to get the full path to the default BEAST2 executable.

Examples

```
library(testthat)

if (is_beast2_installed()) {
  filename <- get_beast2_example_filename("testJukesCantor.xml")
  expect_true(file.exists(filename))

  filename <- get_beast2_example_filename("Primates.nex")
  expect_true(file.exists(filename))

  expect_error(get_beast2_example_filename("abs.ent"))
}
```

```
get_beast2_example_filenames
```

Get a list with the full paths of all BEAST2 example filenames

Description

Get a list with the full paths of all BEAST2 example filenames

Usage

```
get_beast2_example_filenames(beast2_folder = get_default_beast2_folder())
```

Arguments

`beast2_folder` the folder where the BEAST2 is installed. Note that this is not the folder where the BEAST2 executable is installed: the BEAST2 executable is in a sub-folder. Use [get_default_beast2_folder](#) to get the default BEAST2 folder. Use [get_default_beast2_bin_path](#) to get the full path to the default BEAST2 executable.

Value

a list with the full paths of all BEAST2 example filenames

```
get_beast2_main_class_name
```

Get the BEAST2 main class name.

Description

One way to fix the error no main manifest attribute is to specify the main class name.

Usage

```
get_beast2_main_class_name()
```

```
get_beast2_options_filenames
      Extract the filenames from a beast2_options
```

Description

Extract the filenames from a `beast2_options`

Usage

```
get_beast2_options_filenames(beast2_options)
```

Arguments

`beast2_options` a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

beast2_options <- beastier::create_beast2_options()
filenames <- get_beast2_options_filenames(beast2_options)
expect_true(beast2_options$input_filename %in% filenames)
expect_true(beast2_options$output_state_filename %in% filenames)
```

```
get_beast2_version      Get the BEAST2 version
```

Description

Get the BEAST2 version

Usage

```
get_beast2_version(beast2_path = get_default_beast2_path())
```

Arguments

`beast2_path` name of either a BEAST2 binary file (usually simply `beast`) or a BEAST2 jar file (usually has a `.jar` extension). Use [get_default_beast2_bin_path](#) to get the default BEAST binary file's path Use [get_default_beast2_jar_path](#) to get the default BEAST jar file's path

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {
  expect_equal(get_beast2_version(), "2.6.0")
}
```

get_beastier_path	<i>Get the full path of a file in the inst/extdata folder</i>
-------------------	---

Description

Get the full path of a file in the inst/extdata folder

Usage

```
get_beastier_path(filename)
```

Arguments

filename the file's name, without the path

Value

the full path to the filename. Will stop if the file is absent in the inst/extdata folder

Author(s)

Richèl J.C. Bilderbeek

See Also

for more files, use [get_beastier_paths](#)

Examples

```
library(testthat)

expect_true(is.character(get_beastier_path("beast2_example_output.log")))
expect_true(is.character(get_beastier_path("beast2_example_output.trees")))
expect_true(is.character(get_beastier_path("beast2_example_output.xml")))
expect_true(
  is.character(get_beastier_path("beast2_example_output.xml.state"))
)
```

get_beastier_paths *Get the full paths of files in the inst/extdata folder*

Description

Get the full paths of files in the inst/extdata folder

Usage

```
get_beastier_paths(filenamees)
```

Arguments

filenamees the files' names, without the path

Value

the filenamees' full paths. Will stop if a file is absent in the inst/extdata folder

Author(s)

Richèl J.C. Bilderbeek

See Also

for one file, use [get_beastier_path](#)

Examples

```
library(testthat)

filenamees <- get_beastier_paths(
  c(
    "beast2_example_output.log",
    "beast2_example_output.trees",
    "beast2_example_output.xml",
    "beast2_example_output.xml.state"
  )
)

expect_equal(length(filenamees), 4)
expect_true(all(file.exists(filenamees)))
```

```
get_default_beast2_bin_path
```

Get the default BEAST2 binary file (beast, that is) path

Description

Get the default BEAST2 binary file (beast, that is) path

Usage

```
get_default_beast2_bin_path(
  beast2_folder = get_default_beast2_folder(),
  os = rappdirs::app_dir()$os
)
```

Arguments

`beast2_folder` the folder where the BEAST2 is installed. Note that this is not the folder where the BEAST2 executable is installed: the BEAST2 executable is in a sub-folder. Use [get_default_beast2_folder](#) to get the default BEAST2 folder. Use [get_default_beast2_bin_path](#) to get the full path to the default BEAST2 executable.

`os` name of the operating system, must be `unix` (Linux, Mac) or `win` (Windows)

Value

the default BEAST2 binary file's path

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [get_default_beast2_folder](#) to get the default folder in which BEAST2 is installed. Use [install_beast2](#) with default arguments to install BEAST2 to this location.

Examples

```
if (is_beast2_installed() && rappdirs::app_dir()$os == "unix") {
  testit::assert(
    grepl(
      "beast/bin/beast",
      get_default_beast2_bin_path()
    )
  )
}
```

`get_default_beast2_download_url`

Get the default BEAST2 download URL, which depends on the operating system

Description

Get the default BEAST2 download URL, which depends on the operating system

Usage

```
get_default_beast2_download_url(  
  beast2_version = beastier::get_default_beast2_version(),  
  os = rappdirs::app_dir()$os  
)
```

Arguments

`beast2_version` the version of BEAST2. By default, this is the version as returned by [get_default_beast2_version](#)
`os` name of the operating system, must be `unix` (Linux, Mac) or `win` (Windows)

Value

the URL where BEAST2 can be downloaded from

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)  
  
expect_true(  
  grepl(  
    "https://github.com/CompEvol/beast2/releases/download",  
    get_default_beast2_download_url()  
  )  
)  
  
if (rappdirs::app_dir()$os == "unix") {  
  expect_true(  
    grepl(  
      "BEAST.v[0-9\\.]+.Linux.tgz",  
      get_default_beast2_download_url()  
    )  
  )  
}
```

get_default_beast2_download_url_linux

Get the BEAST2 download URL for Linux

Description

Get the BEAST2 download URL for Linux

Usage

```
get_default_beast2_download_url_linux(  
    beast2_version = beastier::get_default_beast2_version()  
)
```

Arguments

beast2_version the version of BEAST2. By default, this is the version as returned by [get_default_beast2_version](#)

Value

the URL where BEAST2 can be downloaded from

Author(s)

Richèl J.C. Bilderbeek

get_default_beast2_download_url_win

Get the BEAST2 download URL for Windows

Description

Get the BEAST2 download URL for Windows

Usage

```
get_default_beast2_download_url_win(  
    beast2_version = beastier::get_default_beast2_version()  
)
```

Arguments

beast2_version the version of BEAST2. By default, this is the version as returned by [get_default_beast2_version](#)

Value

the URL where BEAST2 can be downloaded from

Author(s)

Richèl J.C. Bilderbeek

`get_default_beast2_folder`

Get the path to the folder where this package installs BEAST2 by default

Description

Get the path to the folder where this package installs BEAST2 by default

Usage

```
get_default_beast2_folder()
```

Value

the path to the folder where this package installs BEAST2 by default

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [get_default_beast2_jar_path](#) to get the path to the BEAST2 jar file, when installed by this package Use [install_beast2](#) with default arguments to install BEAST2 to this folder.

Examples

```
print(get_default_beast2_folder())
```

`get_default_beast2_jar_path`

Get the default BEAST2 jar file's path

Description

Get the default BEAST2 jar file's path

Usage

```
get_default_beast2_jar_path(os = rappdirs::app_dir())$os)
```

Arguments

os name of the operating system, must be unix (Linux, Mac) or win (Windows)

Value

the default BEAST2 jar file's path

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [get_default_beast2_folder](#) to get the default folder in which BEAST2 is installed. Use [install_beast2](#) with default arguments to install BEAST2 to this location.

Examples

```
library(testthat)

if (is_beast2_installed() && rappdirs::app_dir()$os == "unix") {
  expect_true(
    grepl(
      "beast/lib/launcher.jar",
      get_default_beast2_jar_path()
    )
  )
}
```

get_default_beast2_path

Get the default BEAST2 path

Description

Get the default BEAST2 path

Usage

```
get_default_beast2_path()
```

Value

the default BEAST2 path

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [get_default_beast2_bin_path](#) to get the default path to the BEAST2 binary file. Use [get_default_beast2_jar_path](#) to get the default path to the BEAST2 jar file. Use [get_default_beast2_folder](#) to get the default folder in which BEAST2 is installed. Use [install_beast2](#) with default arguments to install BEAST2 to this location.

Examples

```
library(testthat)

if (is_beast2_installed()) {
  def_path <- get_default_beast2_path()
  bin_path <- get_default_beast2_bin_path()
  jar_path <- get_default_beast2_jar_path()
  expect_true(def_path == bin_path || def_path == jar_path)
}
```

`get_default_beast2_version`

Get the default BEAST2 version that is used by beastier

Description

Get the default BEAST2 version that is used by beastier

Usage

```
get_default_beast2_version()
```

Value

the BEAST2 version

Author(s)

Richèl J.C. Bilderbeek

Examples

```
get_default_beast2_version()
```

get_default_java_path *Obtains the default path to the Java executable*

Description

Obtains the default path to the Java executable

Usage

```
get_default_java_path(os = rappdirs::app_dir())$os
```

Arguments

os name of the operating system, must be unix (Linux, Mac) or win (Windows)

Value

the default path to the Java executable

Author(s)

Richèl J.C. Bilderbeek

get_duplicate_param_ids

Find duplicate RealParameter IDs

Description

Find duplicate RealParameter IDs

Usage

```
get_duplicate_param_ids(text)
```

Arguments

text the XML as text

Value

a vector of duplicate IDs, will be empty if all IDs are unique

Author(s)

Richèl J.C. Bilderbeek

See Also

to see if all IDs are unique, use [has_unique_ids](#)

Examples

```
line_1 <- "<parameter id=\"RealParameter.1\" ...</parameter>"
line_2 <- "<parameter id=\"RealParameter.2\" ...</parameter>"
testit::assert(
  length(get_duplicate_param_ids(c(line_1, line_2))) == 0)
testit::assert(
  get_duplicate_param_ids(
    c(line_1, line_1)) == c("RealParameter.1")
)
testit::assert(
  get_duplicate_param_ids(
    c(line_2, line_2)) == c("RealParameter.2")
)
```

get_java_version *Get the Java version*

Description

Get the Java version

Usage

```
get_java_version()
```

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {
  expect_silent(get_java_version())
}
```

get_trees_filenames *Get the .trees filenames that BEAST2 will produce*

Description

Get the .trees filenames that BEAST2 will produce

Usage

```
get_trees_filenames(input_filename)
```

Arguments

`input_filename` the name of a BEAST2 input XML file. This file usually has an .xml extension. Use [create_temp_input_filename](#) to create a temporary filename with that extension.

Value

character vector with the names of the .trees files that BEAST2 will produce

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_equal(
  get_trees_filenames(get_beastier_path("2_4.xml")),
  "test_output_0.trees"
)

expect_equal(
  get_trees_filenames(get_beastier_path("anthus_2_4.xml")),
  c("Anthus_nd2.trees", "Anthus_aco.trees")
)
```

`gives_beast2_warning` *Determines if BEAST2 issues a warning when using the BEAST2 XML input file*

Description

Determines if BEAST2 issues a warning when using the BEAST2 XML input file

Usage

```
gives_beast2_warning(  
  filename,  
  verbose = FALSE,  
  beast2_path = get_default_beast2_path()  
)
```

Arguments

<code>filename</code>	name of the BEAST2 XML input file
<code>verbose</code>	if TRUE, additional information is displayed, that is potentially useful in debugging
<code>beast2_path</code>	name of either a BEAST2 binary file (usually simply <code>beast</code>) or a BEAST2 jar file (usually has a <code>.jar</code> extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path

Value

TRUE if the file produces a BEAST2 warning, FALSE if not

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [is_beast2_input_file](#) to check if a file is a valid BEAST2 input file. Use [are_beast2_input_lines](#) to check if the text (for example, as loaded from a file) to be valid BEAST2 input.

Examples

```
library(testthat)  
  
if (is_beast2_installed() &&  
  is_on_ci() &&  
  rappdirs::app_dir()$os == "unix") {  
  
  # This file is OK for BEAST2
```

```
expect_false(  
  gives_beast2_warning(  
    filename = get_beastier_path("2_4.xml")  
  )  
)  
  
# BEAST2 will give a warning on this file  
expect_true(  
  gives_beast2_warning(  
    filename = get_beastier_path("beast2_warning.xml")  
  )  
)  
}
```

has_unique_ids	<i>Determine if the XML text has unique parameter IDs</i>
----------------	---

Description

Determine if the XML text has unique parameter IDs

Usage

```
has_unique_ids(text)
```

Arguments

text the XML as text

Value

TRUE if all parameter IDs are unique, FALSE otherwise

Author(s)

Richèl J.C. Bilderbeek

See Also

to obtain the duplicate parameter IDs, use [get_duplicate_param_ids](#)

Examples

```
library(testthat)  
  
line_1 <- "<parameter id=\"RealParameter.1\" ...</parameter>"  
line_2 <- "<parameter id=\"RealParameter.2\" ...</parameter>"  
expect_true(has_unique_ids(c(line_1, line_2)))  
expect_false(has_unique_ids(c(line_1, line_1)))
```

install_beast2	<i>Install BEAST2 Installs BEAST2 of the default version (see get_default_beast2_version), but another version can also be specified</i>
----------------	--

Description

Install BEAST2 Installs BEAST2 of the default version (see [get_default_beast2_version](#)), but another version can also be specified

Usage

```
install_beast2(
  folder_name = rappdirs::user_data_dir(),
  beast2_version = beastier::get_default_beast2_version(),
  verbose = FALSE,
  os = rappdirs::app_dir()$os
)
```

Arguments

folder_name	name of the folder where the BEAST2 files will be put. The name of the BEAST2 binary file will be at [folder_name]/beast/bin/beast The name of the BEAST2 jar file will be at [folder_name]/beast/lib/launcher.jar
beast2_version	the version of BEAST2. By default, this is the version as returned by get_default_beast2_version
verbose	if TRUE, additional information is displayed, that is potentially useful in debugging
os	name of the operating system, must be unix (Linux, Mac) or win (Windows)

Value

Nothing. Will install BEAST2

Author(s)

Richèl J.C. Bilderbeek

Examples

```
## Not run:
  install_beast2()

## End(Not run)
```

is_alignment	<i>Determines if the input is an alignment of type DNABin</i>
--------------	---

Description

Determines if the input is an alignment of type [DNABin](#)

Usage

```
is_alignment(input)
```

Arguments

input	The input to be tested
-------	------------------------

Value

TRUE or FALSE

Author(s)

Richèl J.C. Bilderbeek

is_beast2_input_file	<i>Is a file a valid BEAST2 input file?</i>
----------------------	---

Description

Is a file a valid BEAST2 input file?

Usage

```
is_beast2_input_file(  
  filename,  
  show_warnings = FALSE,  
  verbose = FALSE,  
  beast2_path = get_default_beast2_path()  
)
```

Arguments

filename	name of the BEAST2 XML input file
show_warnings	if TRUE, warnings will shown
verbose	if TRUE, additional information is displayed, that is potentially useful in debugging
beast2_path	name of either a BEAST2 binary file (usually simply <code>beast</code>) or a BEAST2 jar file (usually has a <code>.jar</code> extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path

Value

TRUE if the file is valid, FALSE if not

Note

this function only works on standard BEAST2 input files: if a BEAST2 input file is modified to use a certain BEAST2 package, this function will label it as an invalid file

Author(s)

Richèl J.C. Bilderbeek

See Also

Use [are_beast2_input_lines](#) to check the lines

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {

  filename <- get_beastier_path("anthus_2_4.xml")
  expect_true(file.exists(filename))
  expect_true(is_beast2_input_file(filename))

  filename <- get_beastier_path("beast2_example_output.log")
  expect_true(file.exists(filename))
  expect_false(is_beast2_input_file(filename))
}
```

is_beast2_installed *Checks if BEAST2 is installed*

Description

Checks if BEAST2 is installed

Usage

```
is_beast2_installed(
  folder_name = get_default_beast2_folder(),
  os = rappdirs::app_dir()$os
)
```

Arguments

folder_name name of the folder where the BEAST2 files are put. The name of the BEAST2 binary file will be at [folder_name]/beast/bin/beast The name of the BEAST2 jar file will be at [folder_name]/beast/lib/launcher.jar

os name of the operating system, must be unix (Linux, Mac) or win (Windows)

Value

TRUE if BEAST2 is installed

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed()) {
  print("BEAST2 is installed")
}
```

is_bin_path *Is the path a path to the BEAST2 binary file? Does not check if the file at that path is present*

Description

Is the path a path to the BEAST2 binary file? Does not check if the file at that path is present

Usage

```
is_bin_path(path)
```


Arguments

path a string to a path

Value

TRUE if the path is a path to a BEAST2 binary file

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed()) {
  expect_true(is_bin_path("beast"))
  expect_true(is_bin_path("BEAST.exe"))
  expect_false(is_bin_path("launcher.jar"))
  expect_true(is_bin_path(get_default_beast2_bin_path()))
  expect_false(is_bin_path(get_default_beast2_jar_path()))
}
```

is_jar_path	<i>Is the path a path to the BEAST2 jar file? Does not check if the file at that path is present</i>
-------------	--

Description

Is the path a path to the BEAST2 jar file? Does not check if the file at that path is present

Usage

```
is_jar_path(path)
```

Arguments

path a string to a path

Value

TRUE if the path is a path to a BEAST2 jar file

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

expect_false(is_jar_path("beast"))
expect_true(is_jar_path("beast.jar"))
expect_true(is_jar_path("launcher.jar"))
expect_false(is_jar_path(get_default_beast2_bin_path()))
expect_true(is_jar_path(get_default_beast2_jar_path()))
```

is_on_appveyor	<i>Determines if the environment is AppVeyor</i>
----------------	--

Description

Determines if the environment is AppVeyor

Usage

```
is_on_appveyor()
```

Value

TRUE if run on AppVeyor, FALSE otherwise

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_on_appveyor()) {
  print("Running on AppVeyor")
}
```

is_on_ci	<i>Determines if the environment is a continuous integration service</i>
----------	--

Description

Determines if the environment is a continuous integration service

Usage

```
is_on_ci()
```

Value

TRUE if run on AppVeyor or Travis CI, FALSE otherwise

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_on_ci()) {  
  print("Running on a continuous integration service")  
}
```

is_on_travis

Determines if the environment is Travis CI

Description

Determines if the environment is Travis CI

Usage

```
is_on_travis()
```

Value

TRUE if run on Travis CI, FALSE otherwise

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_on_travis()) {  
  print("Running on Travis CI")  
}
```

print_beast2_internal_filenames

Print a beast2_internal_filenames as a table

Description

Print a beast2_internal_filenames as a table

Usage

```
print_beast2_internal_filenames(beast2_internal_filenames)
```

Arguments

beast2_internal_filenames

a list of internally used BEAST2 filenames, as created by [create_beast2_internal_filenames](#)

print_beast2_options *Pretty-print a beast2_options*

Description

Pretty-print a beast2_options

Usage

```
print_beast2_options(beast2_options)
```

Arguments

beast2_options a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

remove_file_if_present
Remove a file if it is present

Description

Remove a file if it is present

Usage

```
remove_file_if_present(filename)
```

Arguments

filename name of a file

rename_beast2_options_filenames
Rename the filenames in the BEAST2 options

Description

Rename the filenames in the BEAST2 options

Usage

```
rename_beast2_options_filenames(beast2_options, rename_fun)
```

Arguments

beast2_options a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

rename_fun a function to rename a filename, as can be checked by [check_rename_fun](#). This function should have one argument, which will be a filename or `NA`. The function should [return](#) one filename (when passed one filename) or one `NA` (when passed one `NA`). Example rename functions are:

- [get_remove_dir_fun](#) get a function that removes the directory paths from the filenames, in effect turning these into local files
- [get_replace_dir_fun](#) get a function that replaces the directory paths from the filenames
- [get_remove_hex_fun](#) get a function that removes the hex string from filenames. For example, `tracelog_82c1a522040.log` becomes `tracelog.log`

run_beast2

*Run BEAST2***Description**

Run BEAST2

Usage

```
run_beast2(
  input_filename,
  output_log_filename = "output_log_filename_is_deprecated",
  output_trees_filenames = "output_trees_filenames_is_deprecated",
  output_state_filename = create_temp_state_filename(),
  rng_seed = NA,
  n_threads = NA,
  use_beagle = FALSE,
  overwrite = TRUE,
  beast2_working_dir = "beast2_working_dir_is_deprecated",
  beast2_path = get_default_beast2_path(),
  verbose = FALSE
)
```

Arguments

input_filename the name of a BEAST2 input XML file. This file usually has an .xml extension. Use [create_temp_input_filename](#) to create a temporary filename with that extension.

output_log_filename name of the .log file to create

output_trees_filenames one or more names for .trees file to create. There will be one .trees file created per alignment in the input file. The number of alignments must equal the number of .trees filenames, else an error is thrown. Alignments are sorted alphabetically by their IDs

output_state_filename name of the .xml.state file to create. Use [create_temp_state_filename](#) to create a temporary filename with that extension.

rng_seed the random number generator seed of the BEAST2 run. Must be a non-zero positive integer value or **NA**. If **rng_seed** is **NA**, BEAST2 will pick a random seed

n_threads the number of computational threads to use. Use **NA** to use the BEAST2 default of 1.

use_beagle use BEAGLE if present

overwrite if **TRUE**: overwrite the .log and .trees files if one of these exists. If **FALSE**, BEAST2 will not be started if

- the .log file exists
- the .trees files exist
- the .log file created by BEAST2 exists
- the .trees files created by BEAST2 exist

beast2_working_dir

a folder where BEAST2 can work in isolation. For each BEAST2 run, a new subfolder is created in that folder. Within this folder, BEAST2 is allowed to create all of its output files, without the risk of overwriting existing ones, allowing BEAST2 to run in multiple parallel processes.

beast2_path

name of either a BEAST2 binary file (usually simply `beast`) or a BEAST2 jar file (usually has a `.jar` extension). Use `get_default_beast2_bin_path` to get the default BEAST binary file's path Use `get_default_beast2_jar_path` to get the default BEAST jar file's path

verbose

if TRUE, additional information is displayed, that is potentially useful in debugging

Value

The text sent to STDOUT and STDERR. It will create the file with name `output_state_filenames`

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {

  output_state_filename <- tempfile(fileext = ".xml.state")

  expect_false(file.exists(output_state_filename))

  output <- run_beast2(
    input_filename = get_beastier_path("2_4.xml"),
    output_state_filename = output_state_filename
  )

  expect_true(length(output) > 40)
  expect_true(file.exists(output_state_filename))
}
```

run_beast2_from_options

Run BEAST2

Description

Run BEAST2

Usage

```
run_beast2_from_options(beast2_options = create_beast2_options())
```

Arguments

`beast2_options` a set of BEAST2 options, that are the R equivalent of the BEAST2 command-line options, as can be created by [create_beast2_options](#)

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {

  beast2_options <- create_beast2_options(
    input_filename = get_beastier_path("2_4.xml")
  )

  expect_false(file.exists(beast2_options$output_state_filename))

  output <- run_beast2_from_options(beast2_options)

  expect_true(length(output) > 40)
  expect_true(file.exists(beast2_options$output_state_filename))
}
```

save_lines

Save text (a container of strings) to a file

Description

Save text (a container of strings) to a file

Usage

```
save_lines(filename, lines)
```

Arguments

filename	filename of the file to have the text written to
lines	lines of text to be written to file

Value

Nothing. Will save the lines to file

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

text <- c("hello", "world")
filename <- tempfile(fileext = ".txt")

expect_false(file.exists(filename))
save_lines(filename = filename, lines = text)
expect_true(file.exists(filename))
```

save_nexus_as_fasta *Save a NEXUS file as a FASTA file*

Description

Save a NEXUS file as a FASTA file

Usage

```
save_nexus_as_fasta(nexus_filename, fasta_filename)
```

Arguments

nexus_filename	name of an existing NEXUS file
fasta_filename	name of the FASTA file to be created

uninstall_beast2	<i>Uninstall BEAST2</i>
------------------	-------------------------

Description

Uninstall BEAST2

Usage

```
uninstall_beast2(  
  folder_name = rappdirs::user_data_dir(),  
  os = rappdirs::app_dir()$os  
)
```

Arguments

folder_name	name of the folder where the BEAST2 files are installed. The name of the BEAST2 binary file will be at [folder_name]/beast/bin/beast The name of the BEAST2 jar file will be at [folder_name]/beast/lib/launcher.jar
os	name of the operating system, must be unix (Linux, Mac) or win (Windows)

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)  
  
if (is_beast2_installed() && is_on_ci()) {  
  uninstall_beast2()  
  
  expect_false(is_beast2_installed())  
  
  install_beast2()  
  
  expect_true(is_beast2_installed())  
}
```

update_beastier	<i>Update all beastier dependencies, by installing their latest versions</i>
-----------------	--

Description

Update all beastier dependencies, by installing their latest versions

Usage

```
update_beastier()
```

Author(s)

Richèl J.C. Bilderbeek

upgrade_beast2	<i>Upgrade BEAST2.</i>
----------------	------------------------

Description

Will [stop](#) if BEAST2 is not installed

Usage

```
upgrade_beast2(
  folder_name = rappdirs::user_data_dir(),
  os = rappdirs::app_dir()$os
)
```

Arguments

folder_name	name of the folder where the BEAST2 files will be put. The name of the BEAST2 binary file will be at [folder_name]/beast/bin/beast The name of the BEAST2 jar file will be at [folder_name]/beast/lib/launcher.jar
os	name of the operating system, must be unix (Linux, Mac) or win (Windows)

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed() && is_on_ci()) {
  expect_equal(get_beast2_version(), "2.6.0")
}
```

Index

are_beast2_input_lines, [4](#), [51](#), [55](#)
are_beast2_input_lines_deep, [5](#)
are_beast2_input_lines_fast, [6](#)
are_identical_alignments, [7](#)

beast2_internal_filenames_to_table, [7](#)
beast2_options_to_table, [8](#)
beastier, [8](#), [9](#)
beastier_report, [9](#)

check_beast2, [9](#)
check_beast2_internal_filenames, [10](#),
[11–13](#)
check_beast2_internal_filenames_data_types,
[11](#)
check_beast2_internal_filenames_filenames_differ,
[12](#)
check_beast2_internal_filenames_names,
[12](#)
check_beast2_options, [13](#), [15–17](#)
check_beast2_options_data_types, [15](#)
check_beast2_options_do_not_overwrite_existing_files,
[15](#)
check_beast2_options_filenames_differ,
[16](#)
check_beast2_options_names, [17](#)
check_beast2_optionses, [14](#)
check_beast2_path, [17](#)
check_can_create_file, [18](#)
check_input_filename, [19](#)
check_n_threads, [19](#)
check_os, [20](#)
check_rename_fun, [35](#), [61](#)
check_rng_seed, [21](#)
create_beast2_internal_filenames, [7](#),
[10–12](#), [16](#), [22](#), [33](#), [60](#)
create_beast2_options, [8](#), [13–17](#), [22](#), [23](#),
[33](#), [39](#), [60](#), [61](#), [64](#)
create_beast2_run_cmd, [25](#)
create_beast2_validate_cmd, [26](#)
create_beast2_validate_cmd_bin, [27](#)
create_beast2_validate_cmd_jar, [28](#)
create_beast2_version_cmd, [29](#)
create_beast2_version_cmd_bin, [30](#)
create_beast2_version_cmd_jar, [30](#)
create_temp_input_filename, [19](#), [23](#),
[25–28](#), [31](#), [34](#), [50](#), [62](#)
create_temp_state_filename, [23](#), [32](#), [34](#),
[62](#)

default_params_doc, [32](#)
DNABin, [54](#)
do_minimal_run, [36](#)
get_alignment_ids_from_xml_filename,
[36](#)
get_beast2_example_filename, [37](#)
get_beast2_example_filenames, [38](#)
get_beast2_main_class_name, [38](#)
get_beast2_options_filenames, [39](#)
get_beast2_version, [39](#)
get_beastier_path, [40](#), [41](#)
get_beastier_paths, [40](#), [41](#)
get_default_beast2_bin_path, [4](#), [5](#), [9](#), [17](#),
[24–27](#), [29](#), [30](#), [33](#), [37–39](#), [42](#), [42](#), [47](#),
[51](#), [55](#), [63](#)
get_default_beast2_download_url, [43](#)
get_default_beast2_download_url_linux,
[44](#)
get_default_beast2_download_url_win,
[44](#)
get_default_beast2_folder, [33](#), [37](#), [38](#), [42](#),
[45](#), [46](#), [47](#)
get_default_beast2_jar_path, [4](#), [5](#), [9](#), [17](#),
[24–26](#), [28](#), [29](#), [31](#), [33](#), [39](#), [45](#), [45](#), [47](#),
[51](#), [55](#), [63](#)
get_default_beast2_path, [46](#)
get_default_beast2_version, [33](#), [43](#), [44](#),
[47](#), [53](#)
get_default_java_path, [48](#)

get_duplicate_param_ids, [48](#), [52](#)
get_java_version, [49](#)
get_remove_dir_fun, [35](#), [61](#)
get_remove_hex_fun, [35](#), [61](#)
get_replace_dir_fun, [35](#), [61](#)
get_trees_filenames, [50](#)
gives_beast2_warning, [51](#)

has_unique_ids, [49](#), [52](#)

install_beast2, [42](#), [45–47](#), [53](#)
is_alignment, [54](#)
is_beast2_input_file, [4–6](#), [51](#), [54](#)
is_beast2_installed, [56](#)
is_bin_path, [56](#)
is_jar_path, [57](#)
is_on_appveyor, [58](#)
is_on_ci, [58](#)
is_on_travis, [59](#)

NA, [20](#), [21](#), [23](#), [25](#), [34](#), [35](#), [61](#), [62](#)

print_beast2_internal_filenames, [60](#)
print_beast2_options, [60](#)

remove_file_if_present, [61](#)
rename_beast2_options_filenames, [61](#)
return, [35](#), [61](#)
run_beast2, [62](#)
run_beast2_from_options, [36](#), [64](#)

save_lines, [64](#)
save_nexus_as_fasta, [65](#)
stop, [15](#), [18](#), [19](#), [21](#), [37](#), [67](#)

uninstall_beast2, [66](#)
update_beastier, [67](#)
upgrade_beast2, [67](#)