Package ‘beautier’

March 21, 2018

Title 'BEAUti' from R

Version 1.13

Date 2018-03-16

Author Richel J.C. Bilderbeek <richel@richelbilderbeek.nl>

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

Description 'BEAST2' needs an XML input file.
'BEAUti' is a GUI that does this, requiring active user input.
This R package creates equivalent XML files from function parameters.

License GPL-3 | file LICENSE

LazyData true

RoxygenNote 6.0.1

VignetteBuilder knitr

URL https://github.com/richelbilderbeek/beautier

BugReports https://github.com/richelbilderbeek/beautier/issues

Imports ape, geiger, seqinr, stringr, testit

Suggests devtools, knitr, ggplot2, lintr, rmarkdown, testthat

R topics documented:

are_clock_models *Determine if x consists out of clock_models objects*

Description

Determine if x consists out of clock_models objects

Usage

```
are_clock_models(x)
```

Arguments

`x`	the object to check if it consists out of clock_models objects

Value

TRUE if x, or all elements of x, are clock_model objects

Author(s)

Richel J.C. Bilderbeek

are_equal_xml_files *Determine if XML files result in equal trees*

Description

Determine if XML files result in equal trees

Usage

```
are_equal_xml_files(filename_1, filename_2, section)
```

Arguments

`filename_1`	name of a first XML file
`filename_2`	name of a second XML file
`section`	name of an XML section. Assumes that there is one line that starts with `<section` (excluding whitespace) and one line that is `</section>` (also excluding whitespace)

Value

TRUE if the two sections of the XML files are equal, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

to check for equivalence, use `are_equivalent_xml_files`

`are_equal_xml_lines` *Determine if XML lines result in equal trees*

Description

Determine if XML lines result in equal trees

Usage

```
are_equal_xml_lines(lines_1, lines_2, section)
```

Arguments

`lines_1`	lines of a first XML file
`lines_2`	lines of a second XML file
`section`	name of an XML section. Assumes that there is one line that starts with `<section` (excluding whitespace) and one line that is `</section>` (also excluding whitespace)

Value

TRUE if the two sections of the XML files are equal, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

are_equivalent_xml_files *Determine if XML files result in equivalent trees*

Description

Determine if XML files result in equivalent trees

Usage

```
are_equivalent_xml_files(filename_1, filename_2, section = NA)
```

Arguments

filename_1	name of a first XML file
filename_2	name of a second XML file
section	the name of the XML section, use NA to check the whole file

Value

TRUE if the two XML files result in equivalent trees, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

to check for equality, use are_equal_xml_files

are_equivalent_xml_lines *Determine if XML lines result in equivalent trees*

Description

Determine if XML lines result in equivalent trees

Usage

```
are_equivalent_xml_lines(lines_1, lines_2, section = NA, verbose = FALSE)
```

Arguments

`lines_1` lines of a first XML file

`lines_2` lines of a second XML file

`section` the name of the XML section

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

Value

TRUE if the two XML lines result in equivalent trees, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`are_equivalent_xml_lines_all` *Determine if XML lines result in equivalent trees*

Description

Determine if XML lines result in equivalent trees

Usage

```
are_equivalent_xml_lines_all(lines_1, lines_2, verbose = FALSE)
```

Arguments

`lines_1` lines of a first XML file

`lines_2` lines of a second XML file

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

Value

TRUE if the two XML lines result in equivalent trees, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

are_equivalent_xml_lines_loggers

Determine if XML operator lines result in equivalent trees

Description

Determine if XML operator lines result in equivalent trees

Usage

```
are_equivalent_xml_lines_loggers(lines_1, lines_2, verbose = FALSE)
```

Arguments

`lines_1`	lines of a first XML file
`lines_2`	lines of a second XML file
`verbose`	if TRUE, additional information is displayed, that is potentially useful in debugging

Value

TRUE if the two XML lines result in equivalent trees, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

are_equivalent_xml_lines_operators

Determine if XML operator lines result in equivalent trees

Description

Determine if XML operator lines result in equivalent trees

Usage

```
are_equivalent_xml_lines_operators(lines_1, lines_2, verbose = FALSE)
```

Arguments

`lines_1`	lines of a first XML file
`lines_2`	lines of a second XML file
`verbose`	if TRUE, additional information is displayed, that is potentially useful in debugging

Value

TRUE if the two XML lines result in equivalent trees, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`are_equivalent_xml_lines_section` *Determine if XML lines result in equivalent trees*

Description

Determine if XML lines result in equivalent trees

Usage

```
are_equivalent_xml_lines_section(lines_1, lines_2, section, verbose = FALSE)
```

Arguments

`lines_1`	lines of a first XML file
`lines_2`	lines of a second XML file
`section`	the name of the XML section
`verbose`	if TRUE, additional information is displayed, that is potentially useful in debugging

Value

TRUE if the two XML lines result in equivalent trees, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`are_ids` *Determine if x consists out of IDs*

Description

Determine if x consists out of IDs

Usage

```
are_ids(x)
```

Arguments

`x`	the object to check if it consists out of IDs

Value

TRUE if x, or all elements of x, are IDs

Author(s)

Richel J.C. Bilderbeek

See Also

to check one ID, use `is_id`

`are_init_clock_models` *Determine if x consists out of initialized clock_models objects*

Description

Determine if x consists out of initialized clock_models objects

Usage

```
are_init_clock_models(x)
```

Arguments

`x`	the object to check if it consists out of initialized clock_models objects

Value

TRUE if x, or all elements of x, are initialized clock_model objects

Author(s)

Richel J.C. Bilderbeek

are_init_mrca_priors *Determine if x consists out of initialized mrca_priors objects*

Description

Determine if x consists out of initialized mrca_priors objects

Usage

```
are_init_mrca_priors(x)
```

Arguments

x	the object to check if it consists out of initialized mrca_priors objects

Value

TRUE if x, or all elements of x, are initialized mrca_prior objects

Author(s)

Richel J.C. Bilderbeek

are_init_site_models *Determine if x consists out of initialized site_models objects*

Description

Determine if x consists out of initialized site_models objects

Usage

```
are_init_site_models(x)
```

Arguments

x	the object to check if it consists out of initialized site_models objects

Value

TRUE if x, or all elements of x, are initialized site_model objects

Author(s)

Richel J.C. Bilderbeek

are_init_tree_priors *Determine if x consists out of initialized tree_priors objects*

Description

Determine if x consists out of initialized tree_priors objects

Usage

```
are_init_tree_priors(x)
```

Arguments

`x`	the object to check if it consists out of initialized tree_priors objects

Value

TRUE if x, or all elements of x, are initialized tree_prior objects

Author(s)

Richel J.C. Bilderbeek

are_mrca_align_ids_in_fastas *Determine if the MRCA priors' alignment IDs are present in the FASTA files*

Description

Determine if the MRCA priors' alignment IDs are present in the FASTA files

Usage

```
are_mrca_align_ids_in_fastas(mrca_priors, fasta_filenames)
```

Arguments

`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`
`fasta_filenames`	One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

Author(s)

Richel J.C. Bilderbeek

`are_mrca_priors` *Determine if x consists out of mrca_priors objects*

Description

Determine if x consists out of mrca_priors objects

Usage

```
are_mrca_priors(x)
```

Arguments

`x`	the object to check if it consists out of mrca_priors objects

Value

TRUE if x, or all elements of x, are mrca_prior objects

Author(s)

Richel J.C. Bilderbeek

`are_mrca_taxa_names_in_fastas` *Determine if the MRCA priors' taxa names are present in the FASTA files*

Description

Determine if the MRCA priors' taxa names are present in the FASTA files

Usage

```
are_mrca_taxa_names_in_fastas(mrca_priors, fasta_filenames)
```

Arguments

`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`
`fasta_filenames`	One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

Author(s)

Richel J.C. Bilderbeek

are_mrca_taxa_non_intersecting

Determine if MRCA priors that are monophyletic (i.e., assume no other taxons share a same MRCA) are compatible, that is, there are no taxa that intersect. For example, one MRCA prior with taxon 1 and 2 is incompatible with an MRCA prior with taxon 3 and 4

Description

Determine if MRCA priors that are monophyletic (i.e., assume no other taxons share a same MRCA) are compatible, that is, there are no taxa that intersect. For example, one MRCA prior with taxon 1 and 2 is incompatible with an MRCA prior with taxon 3 and 4

Usage

```
are_mrca_taxa_non_intersecting(mrca_priors)
```

Arguments

`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

Author(s)

Richel J.C. Bilderbeek

are_rln_clock_models

Are the clock models Relaxed Log-Normal clock models?

Description

Are the clock models Relaxed Log-Normal clock models?

Usage

```
are_rln_clock_models(clock_models)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Author(s)

Richel J.C. Bilderbeek

`are_site_models` *Determine if x consists out of site_models objects*

Description

Determine if x consists out of site_models objects

Usage

```
are_site_models(x)
```

Arguments

`x`	the object to check if it consists out of site_models objects

Value

TRUE if x, or all elements of x, are site_model objects

Author(s)

Richel J.C. Bilderbeek

`are_tree_priors` *Determine if x consists out of tree_priors objects*

Description

Determine if x consists out of tree_priors objects

Usage

```
are_tree_priors(x)
```

Arguments

`x`	the object to check if it consists out of tree_priors objects

Value

TRUE if x, or all elements of x, are tree_prior objects

Author(s)

Richel J.C. Bilderbeek

bd_tree_prior_to_xml_prior_distr

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Birth-Death tree prior

Description

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Birth-Death tree prior

Usage

```
bd_tree_prior_to_xml_prior_distr(bd_tree_prior)
```

Arguments

bd_tree_prior	a Birth-Death tree prior, as created by create_bd_tree_prior

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

cbs_tree_prior_to_xml_prior_distr

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Birth-Death tree prior

Description

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Birth-Death tree prior

Usage

```
cbs_tree_prior_to_xml_prior_distr(cbs_tree_prior)
```

Arguments

cbs_tree_prior a Coalescent Bayesian Skyline tree prior, as returned by create_cbs_tree_prior

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

ccp_tree_prior_to_xml_prior_distr

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Coalescent Constant Population tree prior

Description

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Coalescent Constant Population tree prior

Usage

```
ccp_tree_prior_to_xml_prior_distr(ccp_tree_prior)
```

Arguments

ccp_tree_prior a Coalescent Constant Population tree prior, as returned by create_ccp_tree_prior

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

cep_tree_prior_to_xml_prior_distr

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Coalescent Exponential Population tree prior

Description

Creates the tree prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Coalescent Exponential Population tree prior

Usage

```
cep_tree_prior_to_xml_prior_distr(cep_tree_prior)
```

Arguments

cep_tree_prior	a cep tree prior, as returned by create_cep_tree_prior

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

clock_models_to_xml_operators

Create all clock models' operators' XML text

Description

Create all clock models' operators' XML text

Usage

```
clock_models_to_xml_operators(clock_models)
```

Arguments

clock_models a list of one or more clock models, as returned by create_clock_model

Author(s)

Richel J.C. Bilderbeek

---

clock_models_to_xml_prior_distr

Represent the clock models as XML

Description

Represent the clock models as XML

Usage

```
clock_models_to_xml_prior_distr(clock_models)
```

Arguments

clock_models a list of one or more clock models, as returned by create_clock_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

clock_models_to_xml_state

Converts one or more clock models to the `state` section of the XML as text

Description

Converts one or more clock models to the `state` section of the XML as text

Usage

```
clock_models_to_xml_state(clock_models)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Value

lines of XML text, without indentation nor `state` tags

Author(s)

Richel J.C. Bilderbeek

clock_models_to_xml_tracelog

Creates the clock models' XML for the tracelog section

Description

Creates the clock models' XML for the tracelog section

Usage

```
clock_models_to_xml_tracelog(clock_models)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Author(s)

Richel J.C. Bilderbeek

See Also

the complete tracelog section is created by create_beast2_input_tracelog

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

---

clock_model_to_xml_lh_distr

Converts a clock model to the branchRateModel *section of the XML as text*

Description

Converts a clock model to the branchRateModel section of the XML as text

Usage

```
clock_model_to_xml_lh_distr(clock_model, is_first = TRUE,
  is_non_first_shared = TRUE)
```

Arguments

clock_model	a clock model, as returned by create_clock_model
is_first	boolean to indicate if this is the first out of many (one, two or more)
is_non_first_shared	is this clock model not the first of multiple shared ones?

Author(s)

Richel J.C. Bilderbeek

Examples

```
 # <distribution id="posterior" spec="util.CompoundDistribution">
 #     <distribution id="prior" spec="util.CompoundDistribution">
 #     </distribution>
 #     <distribution id="likelihood" ...>
 #       HERE, where the ID of the distribution is 'likelihood'
 #     </distribution>
 # </distribution>
```

clock_model_to_xml_operators *Converts a clock model to the* operators *section of the XML as text*

Description

Converts a clock model to the `operators` section of the XML as text

Usage

```
clock_model_to_xml_operators(clock_model, is_first)
```

Arguments

`clock_model`	a clock model, as returned by `create_clock_model`
`is_first`	boolean to indicate if this is the first out of many (one, two or more)

Author(s)

Richel J.C. Bilderbeek

clock_model_to_xml_prior_distr *Converts a clock model to the* prior *section of the XML as text*

Description

Converts a clock model to the `prior` section of the XML as text

Usage

```
clock_model_to_xml_prior_distr(clock_model, is_first = TRUE)
```

Arguments

`clock_model`	a clock model, as returned by `create_clock_model`
`is_first`	boolean to indicate if this is the first out of many (one, two or more)

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

clock_model_to_xml_state

Converts a clock model to the `state` *section of the XML as text*

Description

Converts a clock model to the `state` section of the XML as text

Usage

```
clock_model_to_xml_state(clock_model)
```

Arguments

`clock_model`	a clock model, as returned by `create_clock_model`

Value

lines of XML text, without indentation nor `state` tags

Author(s)

Richel J.C. Bilderbeek

clock_model_to_xml_tracelog

Creates the clock model's XML for the tracelog section

Description

Creates the clock model's XML for the tracelog section

Usage

```
clock_model_to_xml_tracelog(clock_model, is_first)
```

Arguments

`clock_model`	a clock model, as returned by `create_clock_model`
`is_first`	boolean to indicate if this is the first out of many (one, two or more)

Author(s)

Richel J.C. Bilderbeek

See Also

all clock models' tracelog section is created by `clock_model_to_xml_tracelog`

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

`compare_lines` *Internal debug function to compare the actually created lines to expected lines.*

Description

Internal debug function to compare the actually created lines to expected lines.

Usage

```
compare_lines(lines, expected, section = NA)
```

Arguments

`lines`	the created lines
`expected`	the expected/goal/target lines
`section`	the XML section. Leave at NA to compare all lines

`count_trailing_spaces` *Count the number of spaces before the first character*

Description

Count the number of spaces before the first character

Usage

```
count_trailing_spaces(line)
```

Arguments

`line`	line of text

Value

the number of spaces before the first character

Author(s)

Richel J.C. Bilderbeek

`create_alpha_param` *Create a parameter called alpha*

Description

Create a parameter called alpha

Usage

```
create_alpha_param(id = NA, estimate = FALSE, value = 0)
```

Arguments

`id`	the parameter's ID
`estimate`	TRUE if this parameter alpha be estimated by BEAST2, FALSE otherwise
`value`	value of the parameter

Value

a parameter called alpha

Note

this parameter is used in a beta distribution (as returned by create_beta_distr) and gamma distribution (as returned by create_gamma_distr) and inverse gamma distribution (as returned by create_inv_gamma_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
alpha_param <- create_alpha_param()

# Use the parameter in a distribution
beta_distr <- create_beta_distr(
  alpha = alpha_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_alpha_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = beta_distr
  )
)
testit::assert(file.exists("create_alpha_param.xml"))
```

create_bd_tree_prior *Create a Birth-Death tree prior*

Description

Create a Birth-Death tree prior

Usage

```
create_bd_tree_prior(
  id = NA,
  birth_rate_distr = create_uniform_distr(),
  death_rate_distr = create_uniform_distr()
)
```

Arguments

id the ID of the alignment

birth_rate_distr
the birth rate distribution, as created by a create_distr function

death_rate_distr
the death rate distribution, as created by a create_distr function

Value

a Birth-Death tree_prior

Author(s)

Richel J.C. Bilderbeek

See Also

An alignment ID can be extracted from its FASTA filesname using get_id

Examples

```
bd_tree_prior <- create_bd_tree_prior()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_bd_tree_prior.xml",
  tree_priors = bd_tree_prior
)
testit::assert(file.exists("create_bd_tree_prior.xml"))

bd_tree_prior_exp <- create_bd_tree_prior(
  birth_rate_distr = create_exp_distr()
)

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_bd_tree_prior_exp.xml",
  tree_priors = bd_tree_prior_exp
)
testit::assert(file.exists("create_bd_tree_prior_exp.xml"))
```

---

create_beast2_input *Create a BEAST2 XML input text*

Description

Create a BEAST2 XML input text

Usage

```
create_beast2_input(input_filenames, site_models = create_jc69_site_models(ids
  = get_ids(input_filenames)), clock_models = create_strict_clock_models(ids =
  get_ids(input_filenames)), tree_priors = create_yule_tree_priors(ids =
  get_ids(input_filenames)), mrca_priors = NA, mcmc = create_mcmc(),
  misc_options = create_misc_options(), posterior_crown_age = NA)
```

Arguments

`input_filenames`
: One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`site_models`
: one or more site models, as returned by `create_site_model`

`clock_models`
: a list of one or more clock models, as returned by `create_clock_model`

`tree_priors`
: one or more tree priors, as returned by `create_tree_prior`

`mrca_priors`
: a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

`mcmc`
: one mcmc object, as returned by `create_mcmc`

`misc_options`
: one misc_options object, as returned by `create_misc_options`

`posterior_crown_age`
: the crown age the posteriors' phylogenies will be fixed at. If NA, crown age is estimated by BEAST2.

Author(s)

Richel J.C. Bilderbeek

See Also

`create_beast2_input_file` shows more examples

Examples

```
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "my_beast.xml"
)
```

create_beast2_input_1_12

Create a BEAST2 XML input text, interface of v1.12

Description

Create a BEAST2 XML input text, interface of v1.12

Usage

```
create_beast2_input_1_12(input_filenames,
  site_models = create_jc69_site_models(ids = get_ids(input_filenames)),
  clock_models = create_strict_clock_models(ids = get_ids(input_filenames)),
  tree_priors = create_yule_tree_priors(ids = get_ids(input_filenames)),
  mcmc = create_mcmc(), misc_options = create_misc_options(),
  fixed_crown_ages = rep(FALSE, times = length(input_filenames)),
  initial_phylogenies = rep(NA, length(input_filenames)))
```

Arguments

`input_filenames`
: One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`site_models`
: one or more site models, as returned by `create_site_model`

`clock_models`
: a list of one or more clock models, as returned by `create_clock_model`

`tree_priors`
: one or more tree priors, as returned by `create_tree_prior`

`mcmc`
: one mcmc object, as returned by `create_mcmc`

`misc_options`
: one misc_options object, as returned by `create_misc_options`

`fixed_crown_ages`
: one or more booleans to determine if the phylogenies' crown ages are fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, 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 NA will result in BEAST2 using a random phylogeny. Else the phylogeny is assumed to be of class ape::phylo.

Author(s)

Richel J.C. Bilderbeek

See Also

`create_beast2_input_file` shows more examples

Examples

```
create_beast2_input_file_1_12(
  input_filenames = get_fasta_filename(),
  "my_beast.xml"
)
```

---

create_beast2_input_beast

Creates the beast section of a BEAST2 XML parameter file

Description

Creates the beast section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_beast(input_filenames,
  site_models = create_jc69_site_models(ids = get_ids(input_filenames)),
  clock_models = create_strict_clock_models(ids = get_ids(input_filenames)),
  tree_priors = create_yule_tree_priors(ids = get_ids(input_filenames)),
  mrca_priors = NA, mcmc = create_mcmc(),
  misc_options = create_misc_options(), fixed_crown_ages = rep(FALSE, times
  = length(input_filenames)), initial_phylogenies = rep(NA,
  length(input_filenames)))
```

Arguments

`input_filenames`
One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`site_models` one or more site models, as returned by `create_site_model`

`clock_models` a list of one or more clock models, as returned by `create_clock_model`

`tree_priors` one or more tree priors, as returned by `create_tree_prior`

`mrca_priors` a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

`mcmc` one mcmc object, as returned by `create_mcmc`

`misc_options` one misc_options object, as returned by `create_misc_options`

`fixed_crown_ages`
one or more booleans to determine if the phylogenies' crown ages are fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, 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 NA will result in BEAST2 using a random phylogeny. Else the phylogeny is assumed to be of class ape::phylo.

Author(s)

Richel J.C. Bilderbeek

`create_beast2_input_data` *Creates the data section of a BEAST2 XML parameter file*

Description

Creates the data section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_data(input_filenames,
  misc_options = create_misc_options())
```

Arguments

`input_filenames`
: One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`misc_options`
: one misc_options object, as returned by `create_misc_options`

Author(s)

Richel J.C. Bilderbeek

`create_beast2_input_data_sequences` *Creates the data section of a BEAST2 XML parameter file*

Description

Creates the data section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_data_sequences(input_fasta_filename,
  nucleotides_uppercase = FALSE)
```

Arguments

`input_fasta_filename`
: one FASTA filename

`nucleotides_uppercase`
: are the nucleotides written in uppercase? Yes if TRUE, no if FALSE

Note

this function is not intended for regular use, thus its long name length is accepted

Author(s)

Richel J.C. Bilderbeek

---

create_beast2_input_distr

Creates the distribution section of a BEAST2 XML parameter file.

Description

Creates the distribution section of a BEAST2 XML parameter file.

Usage

```
create_beast2_input_distr(site_models, clock_models, tree_priors,
  mrca_priors = NA)
```

Arguments

site_models	one or more site models, as returned by create_site_model
clock_models	a list of one or more clock models, as returned by create_clock_model
tree_priors	one or more tree priors, as returned by create_tree_prior
mrca_priors	a list of one or more Most Recent Common Ancestor priors, as returned by create_mrca_prior

Note

this function is not intended for regular use, thus its long name length is accepted

Author(s)

Richel J.C. Bilderbeek

See Also

create_beast2_input

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

create_beast2_input_distr_lh

Creates the likelihood section in the distribution section of a BEAST2 XML parameter file

Description

Creates the likelihood section in the distribution section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_distr_lh(site_models, clock_models, mrca_priors = NA)
```

Arguments

`site_models` one or more site models, as returned by `create_site_model`

`clock_models` a list of one or more clock models, as returned by `create_clock_model`

`mrca_priors` a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

Note

this function is not intended for regular use, thus its long name length is accepted

Author(s)

Richel J.C. Bilderbeek

See Also

this function is called by `create_beast2_input_distr`, together with `create_beast2_input_distr_prior`

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#     </distribution>
#     <distribution id="likelihood" ...>
#       HERE, where the ID of the distribution is 'likelihood'
#     </distribution>
# </distribution>
```

create_beast2_input_distr_prior

Creates the prior section in the distribution section of a BEAST2 XML parameter file

Description

Creates the prior section in the distribution section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_distr_prior(site_models, clock_models, tree_priors,
  mrca_priors = NA)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`
`clock_models`	a list of one or more clock models, as returned by `create_clock_model`
`tree_priors`	one or more tree priors, as returned by `create_tree_prior`
`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

Note

this function is not intended for regular use, thus its long name length is accepted

Author(s)

Richel J.C. Bilderbeek

See Also

this function is called by `create_beast2_input_distr`, together with `create_beast2_input_distr_lh`

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

create_beast2_input_file

Create a BEAST2 input file

Description

Create a BEAST2 input file

Usage

```
create_beast2_input_file(input_filenames, output_filename,
  site_models = create_jc69_site_models(ids = get_ids(input_filenames)),
  clock_models = create_strict_clock_models(ids = get_ids(input_filenames)),
  tree_priors = create_yule_tree_priors(ids = get_ids(input_filenames)),
  mrca_priors = NA, mcmc = create_mcmc(), posterior_crown_age = NA)
```

Arguments

`input_filenames`
One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`output_filename`
Name of the XML parameter file created by this function. BEAST2 uses this file as input.

`site_models` one or more site models, as returned by `create_site_model`

`clock_models` a list of one or more clock models, as returned by `create_clock_model`

`tree_priors` one or more tree priors, as returned by `create_tree_prior`

`mrca_priors` a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

`mcmc` one mcmc object, as returned by `create_mcmc`

`posterior_crown_age`
the crown age the posteriors' phylogenies will be fixed at. If NA, crown age is estimated by BEAST2.

Author(s)

Richel J.C. Bilderbeek

See Also

See `create_site_model` for examples with different site models. See `create_clock_model` for examples with clock models. See `create_tree_prior` for examples with different tree priors. See `create_mcmc` for examples with a different MCMC setup.

Examples

```
# The file created by beautier, a BEAST2 input file
output_filename <- "create_beast2_input_file.xml"

# Birth-Death tree prior, crown age is estimated
create_beast2_input_file(
  get_fasta_filename(),
  output_filename
)
testthat::expect_true(file.exists(output_filename))

# The file created by beautier, a BEAST2 input file
output_filename_fixed <- "create_beast2_input_file_fixed.xml"

# Birth-Death tree prior, crown age is fixed at 15 time units
create_beast2_input_file(
  get_fasta_filename(),
  output_filename_fixed,
  posterior_crown_age = 15
)
testthat::expect_true(file.exists(output_filename_fixed))
```

create_beast2_input_file_1_12

Create a BEAST2 input file, interface of v1.12. This interface is obsoleted, use create_beast2_input_file *instead*

Description

Create a BEAST2 input file, interface of v1.12. This interface is obsoleted, use create_beast2_input_file instead

Usage

```
create_beast2_input_file_1_12(input_filenames, output_filename,
  site_models = create_jc69_site_models(ids = get_ids(input_filenames)),
  clock_models = create_strict_clock_models(ids = get_ids(input_filenames)),
  tree_priors = create_yule_tree_priors(ids = get_ids(input_filenames)),
  mcmc = create_mcmc(), fixed_crown_ages = rep(FALSE,
  length(input_filenames)), initial_phylogenies = rep(NA,
  length(input_filenames)))
```

Arguments

input_filenames
: One or more FASTA filenames. Use get_fasta_filename to obtain a testing FASTA filename.

output_filename
: Name of the XML parameter file created by this function. BEAST2 uses this file as input.

site_models
: one or more site models, as returned by create_site_model

clock_models
: a list of one or more clock models, as returned by create_clock_model

tree_priors
: one or more tree priors, as returned by create_tree_prior

mcmc
: one mcmc object, as returned by create_mcmc

fixed_crown_ages
: one or more booleans to determine if the phylogenies' crown ages are fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, 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 NA will result in BEAST2 using a random phylogeny. Else the phylogeny is assumed to be of class ape::phylo.

Author(s)

Richel J.C. Bilderbeek

See Also

See create_site_model for examples with different site models. See create_clock_model for examples with clock models. See create_tree_prior for examples with different tree priors. See create_mcmc for examples with a different MCMC setup. See fasta_to_phylo for examples with a fixed crown age

Examples

```
# The file created by beautier, a BEAST2 input file
output_filename_fixed <- "create_beast2_input_file_1_12.xml"

# Birth-Death tree prior, crown age is fixed at 15 time units
create_beast2_input_file_1_12(
  input_filenames = get_fasta_filename(),
  output_filename = output_filename_fixed,
  fixed_crown_ages = TRUE,
  initial_phylogenies = beautier::fasta_to_phylo(
    fasta_filename = get_fasta_filename(),
    crown_age = 15)
)
testthat::expect_true(file.exists(output_filename_fixed))
```

create_beast2_input_init *Creates the map section of a BEAST2 XML parameter file*

Description

Creates the map section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_init(ids, initial_phylogenies)
```

Arguments

`ids` one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with `get_ids`)

`initial_phylogenies` one or more MCMC chain's initial phylogenies. Each one set to NA will result in BEAST2 using a random phylogeny. Else the phylogeny is assumed to be of class ape::phylo.

Author(s)

Richel J.C. Bilderbeek

create_beast2_input_loggers *Creates the two logger sections of a BEAST2 XML parameter file*

Description

Creates the two logger sections of a BEAST2 XML parameter file

Usage

```
create_beast2_input_loggers(ids, site_models = create_jc69_site_models(ids =
  ids), clock_models = create_strict_clock_models(ids = ids),
  tree_priors = create_yule_tree_priors(ids = ids), mrca_priors = NA)
```

Arguments

ids	one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with get_ids)
site_models	one or more site models, as returned by create_site_model
clock_models	a list of one or more clock models, as returned by create_clock_model
tree_priors	one or more tree priors, as returned by create_tree_prior
mrca_priors	a list of one or more Most Recent Common Ancestor priors, as returned by create_mrca_prior

Author(s)

Richel J.C. Bilderbeek

create_beast2_input_map

Creates the map section of a BEAST2 XML parameter file

Description

Creates the map section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_map()
```

Author(s)

Richel J.C. Bilderbeek

create_beast2_input_operators

Creates the operators section of a BEAST2 XML parameter file

Description

Creates the operators section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_operators(site_models, clock_models, tree_priors,
  fixed_crown_ages = rep(FALSE, length(site_models)), mrca_priors = NA)
```

Arguments

site_models one or more site models, as returned by create_site_model

clock_models a list of one or more clock models, as returned by create_clock_model

tree_priors one or more tree priors, as returned by create_tree_prior

fixed_crown_ages
one or more booleans to determine if the phylogenies' crown ages are fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, the crown age is fixed to the crown age of the initial phylogeny.

mrca_priors a list of one or more Most Recent Common Ancestor priors, as returned by create_mrca_prior

Note

this function is not intended for regular use, thus its long name length is accepted

Author(s)

Richel J.C. Bilderbeek

---

create_beast2_input_run

Creates the state section of a BEAST2 XML parameter file

Description

Creates the state section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_run(ids, site_models = create_jc69_site_models(ids = ids),
  clock_models = create_strict_clock_models(ids = ids),
  tree_priors = create_yule_tree_priors(ids = ids), mrca_priors = NA,
  mcmc = create_mcmc(), fixed_crown_ages = rep(FALSE, times = length(ids)),
  initial_phylogenies = rep(NA, length(ids)))
```

Arguments

ids the IDs of the alignments (can be extracted from their FASTA filesnames using get_ids)

site_models one or more site models, as returned by create_site_model

clock_models a list of one or more clock models, as returned by create_clock_model

tree_priors one or more tree priors, as returned by create_tree_prior

mrca_priors a list of one or more Most Recent Common Ancestor priors, as returned by create_mrca_prior

`mcmc`	one mcmc object, as returned by `create_mcmc`
`fixed_crown_ages`	one or more booleans to determine if the phylogenies' crown ages are fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, the crown age is fixed to the crown age of the initial phylogeny.
`initial_phylogenies`	initial phylogenies, can be NAs if random phylogenies are desired

Author(s)

Richel J.C. Bilderbeek

create_beast2_input_screenlog

Creates the screenlog section of the logger section of a BEAST2 XML parameter file

Description

Creates the screenlog section of the logger section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_screenlog()
```

Value

the XML text

Author(s)

Richel J.C. Bilderbeek

create_beast2_input_state

Creates the state section of a BEAST2 XML parameter file

Description

Creates the state section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_state(site_models, clock_models, tree_priors,
  initial_phylogenies, mrca_priors = NA)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`
`clock_models`	a list of one or more clock models, as returned by `create_clock_model`
`tree_priors`	one or more tree priors, as returned by `create_tree_prior`
`initial_phylogenies`	initial phylogenies, can be NAs if random phylogenies are desired
`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

Note

this function is not intended for regular use, thus its long name length is accepted

Author(s)

Richel J.C. Bilderbeek

---

create_beast2_input_tracelog

Creates the tracelog section of the logger section of a BEAST2 XML parameter file

---

Description

Creates the tracelog section of the logger section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_tracelog(ids, site_models = create_jc69_site_models(ids =
  ids), clock_models = create_strict_clock_models(ids = ids),
  tree_priors = create_yule_tree_priors(ids = ids), mrca_priors = NA)
```

Arguments

`ids`	one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with `get_ids`)
`site_models`	one or more site models, as returned by `create_site_model`
`clock_models`	a list of one or more clock models, as returned by `create_clock_model`
`tree_priors`	one or more tree priors, as returned by `create_tree_prior`
`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

Author(s)

Richel J.C. Bilderbeek

`create_beast2_input_treelogs`

Creates the tracelog section of the logger section of a BEAST2 XML parameter file

Description

Creates the tracelog section of the logger section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_treelogs(clock_models)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Author(s)

Richel J.C. Bilderbeek

`create_beast2_input_xml`

Creates the xml section of a BEAST2 XML parameter file

Description

Creates the xml section of a BEAST2 XML parameter file

Usage

```
create_beast2_input_xml()
```

Author(s)

Richel J.C. Bilderbeek

create_beta_distr *Create a beta distribution*

Description

Create a beta distribution

Usage

```
create_beta_distr(id = NA, alpha = create_alpha_param(),
  beta = create_beta_param())
```

Arguments

id	the distribution's ID
alpha	the alpha shape parameter, as returned by create_alpha_param. The value of alpha must be at least 0.0.
beta	the beta shape parameter, as returned by create_beta_param. The value of beta must be at least 1.0.

Value

a beta distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
beta_distr <- create_beta_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_beta_distr.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = beta_distr
  )
)
testit::assert(file.exists("create_beta_distr.xml"))
```

create_beta_param *Create a parameter called beta*

Description

Create a parameter called beta

Usage

```
create_beta_param(id = NA, estimate = FALSE, value = 1)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter alpha be estimated by BEAST2, FALSE otherwise
value	value of the parameter

Value

a parameter called beta

Note

this parameter is used in a beta distribution (as returned by create_beta_distr) and gamma distribution (as returned by create_gamma_distr) and inverse gamma distribution (as returned by create_inv_gamma_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
beta_param <- create_beta_param()

# Use the parameter in a distribution
gamma_distr <- create_gamma_distr(
  beta = beta_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_beta_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = gamma_distr
  )
)
testit::assert(file.exists("create_beta_param.xml"))
```

create_cbs_tree_prior *Create a Coalescent Bayesian Skyline tree prior*

Description

Create a Coalescent Bayesian Skyline tree prior

Usage

```
create_cbs_tree_prior(id = NA, group_sizes_dimension = 5)
```

Arguments

id	an alignment's IDs. An ID can be extracted from its FASTA filesname with get_ids)
group_sizes_dimension	the group sizes' dimension, as used by the CBS tree prior (see create_cbs_tree_prior)

Value

a Coalescent Bayesian Skyline tree_prior

Author(s)

Richel J.C. Bilderbeek

See Also

An alignment ID can be extracted from its FASTA filesname using get_id

Examples

```
  cbs_tree_prior <- create_cbs_tree_prior()

  create_beast2_input_file(
    input_filenames = get_beautier_path("test_output_6.fas"),
    "create_cbs_tree_prior.xml",
    tree_priors = cbs_tree_prior
  )
  testit::assert(file.exists("create_cbs_tree_prior.xml"))
```

create_ccp_tree_prior *Create a Coalescent Constant Population tree prior*

Description

Create a Coalescent Constant Population tree prior

Usage

```
create_ccp_tree_prior(id = NA,
  pop_size_distr = beautier::create_one_div_x_distr())
```

Arguments

id	the ID of the alignment
pop_size_distr	the population distribution, as created by a create_distr function

Value

a Coalescent Constant Population tree_prior

Author(s)

Richel J.C. Bilderbeek

See Also

An alignment ID can be extracted from its FASTA filesname using get_id

Examples

```
ccp_tree_prior <- create_ccp_tree_prior()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_ccp_tree_prior.xml",
  tree_priors = ccp_tree_prior
)
testit::assert(file.exists("create_ccp_tree_prior.xml"))
```

create_cep_tree_prior *Create a Coalescent Exponential Population tree prior*

Description

Create a Coalescent Exponential Population tree prior

Usage

```
create_cep_tree_prior(id = NA, pop_size_distr = create_one_div_x_distr(),
  growth_rate_distr = create_laplace_distr())
```

Arguments

`id`	the ID of the alignment
`pop_size_distr`	the population distribution, as created by a `create_distr` function
`growth_rate_distr`	the growth rate distribution, as created by a `create_distr` function

Value

a Coalescent Exponential Population tree_prior

Author(s)

Richel J.C. Bilderbeek

See Also

An alignment ID can be extracted from its FASTA filesname using `get_id`

Examples

```
cep_tree_prior <- create_cep_tree_prior()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_cep_tree_prior.xml",
  tree_priors = cep_tree_prior
)
testit::assert(file.exists("create_cep_tree_prior.xml"))
```

create_clock_model *General function to create a clock model*

Description

General function to create a clock model

Usage

```
create_clock_model(name, id, ...)
```

Arguments

name	the clock model name. Valid names can be found in get_clock_model_names
id	a clock model's ID
...	specific clock model parameters

Value

a valid clock model

Note

Prefer using the named function create_rln_clock_model and create_strict_clock_model

Author(s)

Richel J.C. Bilderbeek

See Also

An alignment ID can be extracted from its FASTA filesname using get_id. For more examples about creating a relaxed log-normal clock model, see create_rln_clock_model. For more examples about creating a strict clock model, see create_strict_clock_model.

Examples

```
rln_clock_model <- create_rln_clock_model()

create_beast2_input_file(
  get_fasta_filename(),
  "create_clock_model_rln.xml",
  clock_models = rln_clock_model
)
testit::assert(file.exists("create_clock_model_rln.xml"))

strict_clock_model <- create_strict_clock_model()
```

```
create_beast2_input_file(
  get_fasta_filename(),
  "create_clock_model_strict.xml",
  clock_models = strict_clock_model
)
testit::assert(file.exists("create_clock_model_strict.xml"))
```

create_clock_models *Creates all supported clock models, which is just a list of the types returned by create_rln_clock_model, and create_strict_clock_model*

Description

Creates all supported clock models, which is just a list of the types returned by create_rln_clock_model, and create_strict_clock_model

Usage

```
create_clock_models()
```

Value

a list of site_models

Author(s)

Richel J.C. Bilderbeek

Examples

```
clock_models <- beautier:::create_clock_models()
testit::assert(beautier:::is_rln_clock_model(clock_models[[1]]))
testit::assert(beautier:::is_strict_clock_model(clock_models[[2]]))
```

create_clock_rate_param *Create a parameter called clock_rate, as needed by create_strict_clock_model*

Description

Create a parameter called clock_rate, as needed by create_strict_clock_model

Usage

```
create_clock_rate_param(value = "1.0", estimate = FALSE, id = NA)
```

Arguments

`value`	value of the parameter
`estimate`	TRUE if this parameter is estimated by BEAST2, FALSE otherwise
`id`	the alignment id

Value

a parameter called rate

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
clock_rate_param <- create_clock_rate_param(
  id = "anthus_aco", estimate = FALSE, value = 1.0
)

# Use the parameter in a clock model
strict_clock_model <- create_strict_clock_model(
  clock_rate_param = clock_rate_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_clock_rate_param.xml",
  clock_models = strict_clock_model
)
testit::assert(file.exists("create_clock_rate_param.xml"))
```

`create_distr` *General function to create a distribution.*

Description

General function to create a distribution.

Usage

```
create_distr(name, id, ...)
```

Arguments

name	the distribution name. Valid names can be found in get_distr_names
id	the distribution's ID
...	specific distribution parameters

Value

a distribution

Note

Prefer using the named functions create_beta_distr, create_exp_distr, create_gamma_distr, create_inv_gamma_distr, create_laplace_distr, create_log_normal_distr, create_normal_distr, create_one_div_x_distr, create_poisson_distr and create_uniform_distr

See create_beta_distr, create_exp_distr, create_gamma_distr, create_inv_gamma_distr, create_laplace_distr, create_log_normal_distr, create_normal_distr, create_one_div_x_distr, create_poisson_distr and create_uniform_distr for examples how to use those distributions

Author(s)

Richel J.C. Bilderbeek

Examples

```
# Use any distribution
distr <- create_beta_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_distr.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = distr
  )
)
testit::assert(file.exists("create_distr.xml"))
```

---

create_exp_distr *Create an exponential distribution*

Description

Create an exponential distribution

Usage

```
create_exp_distr(id = NA, mean = create_mean_param(value = 1))
```

Arguments

id	the distribution's ID
mean	the mean parameter, as returned by create_mean_param

Value

an exponential distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
exp_distr <- create_exp_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "my_beast.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = exp_distr
  )
)
testit::assert(file.exists("my_beast.xml"))
```

create_gamma_distr *Create a gamma distribution*

Description

Create a gamma distribution

Usage

```
create_gamma_distr(id = NA, alpha = create_alpha_param(id = NA, estimate =
  FALSE, value = "0.5396"), beta = create_beta_param(id = NA, estimate =
  FALSE, value = "0.3819"))
```

Arguments

id	the distribution's ID
alpha	the alpha shape parameter, as returned by create_alpha_param
beta	the beta shape parameter, as returned by create_beta_param

Value

a gamma distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_distr` shows an overview of all supported distributions

Examples

```
gamma_distr <- create_gamma_distr(
   alpha = create_alpha_param(value = 0.05),
   beta = create_beta_param(value = 10.0)
)

gtr_site_model <- create_gtr_site_model(
  rate_ac_prior_distr = gamma_distr
)

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_gamma_distr.xml",
  site_model = gtr_site_model
)
testit::assert(file.exists("create_gamma_distr.xml"))
```

---

create_gamma_site_model

Create a gamma site model, part of a site model

Description

Create a gamma site model, part of a site model

Usage

```
create_gamma_site_model(gamma_cat_count = "0", gamma_shape = "1.0",
  prop_invariant = "0.0", gamma_shape_prior_distr = create_exp_distr(id =
  NA, mean = create_mean_param(id = 0, value = "1.0")),
  freq_equilibrium = "estimated")
```

Arguments

`gamma_cat_count`
: the number of gamma categories, must be an integer with value zero or more

`gamma_shape`
: gamma curve shape parameter

`prop_invariant`
: the proportion invariant, must be a value from 0.0 to 1.0

`gamma_shape_prior_distr`
: the distribution of the gamma shape prior

`freq_equilibrium`
: the frequency in which the rates are at equilibrium are either `estimated`, `empirical` or `all_equal`. `get_freq_equilibrium_names` returns the possible values for `freq_equilibrium`

Value

a gamma site model

Note

Use `create_site_model` to create a site model that has both a gamma site model and substitution model

Author(s)

Richel J.C. Bilderbeek

Examples

```
gamma_site_model <- create_gamma_site_model(prop_invariant = 0.5)

site_model <- create_hky_site_model(gamma_site_model = gamma_site_model)

create_beast2_input_file(
  get_fasta_filename(),
  "create_gamma_site_model.xml",
  site_model = site_model
)
testit::assert(file.exists("create_gamma_site_model.xml"))
```

`create_gtr_site_model` *Create a GTR site model*

Description

Create a GTR site model

Usage

```
create_gtr_site_model(id = NA, gamma_site_model = create_gamma_site_model(),
  rate_ac_prior_distr = create_gamma_distr(alpha = create_alpha_param(value =
  "0.05"), beta = create_beta_param(value = "10.0")),
  rate_ag_prior_distr = create_gamma_distr(alpha = create_alpha_param(value =
  "0.05"), beta = create_beta_param(value = "20.0")),
  rate_at_prior_distr = create_gamma_distr(alpha = create_alpha_param(value =
  "0.05"), beta = create_beta_param(value = "10.0")),
  rate_cg_prior_distr = create_gamma_distr(alpha = create_alpha_param(value =
  "0.05"), beta = create_beta_param(value = "10.0")),
  rate_gt_prior_distr = create_gamma_distr(alpha = create_alpha_param(value =
  "0.05"), beta = create_beta_param(value = "10.0")),
  rate_ac_param = create_rate_ac_param(),
  rate_ag_param = create_rate_ag_param(),
  rate_at_param = create_rate_at_param(),
  rate_cg_param = create_rate_cg_param(),
  rate_ct_param = create_rate_ct_param(estimate = FALSE),
  rate_gt_param = create_rate_gt_param(), freq_equilibrium = "estimated")
```

Arguments

`id` the IDs of the alignment (can be extracted from the FASTA filesname using `get_id`)

`gamma_site_model` a gamma site model, as created by `create_gamma_site_model`

`rate_ac_prior_distr` the AC rate prior distribution, as returned by `create_distr`)

`rate_ag_prior_distr` the AG rate prior distribution, as returned by `create_distr`)

`rate_at_prior_distr` the AT rate prior distribution, as returned by `create_distr`)

`rate_cg_prior_distr` the CG rate prior distribution, as returned by `create_distr`)

`rate_gt_prior_distr` the GT rate prior distribution, as returned by `create_distr`)

`rate_ac_param` the 'rate AC' parameter, as returned by `create_rate_ac_param`)

`rate_ag_param` the 'rate AG' parameter, as returned by `create_rate_ag_param`)

`rate_at_param` the 'rate AT' parameter, as returned by `create_rate_at_param`)

`rate_cg_param` the 'rate CG' parameter, as returned by `create_rate_cg_param`)

`rate_ct_param` the 'rate CT' parameter, as returned by `create_rate_ct_param`)

`rate_gt_param` the 'rate GT' parameter, as returned by `create_rate_gt_param`)

`freq_equilibrium` the frequency in which the rates are at equilibrium are either `estimated`, `empirical` or `all_equal`. `get_freq_equilibrium_names` returns the possible values for `freq_equilibrium`

Value

a GTR site_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
gtr_site_model <- create_gtr_site_model()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_gtr_site_model.xml",
  site_models = gtr_site_model
)
testit::assert(file.exists("create_gtr_site_model.xml"))
```

`create_hky_site_model` *Create an HKY site model*

Description

Create an HKY site model

Usage

```
create_hky_site_model(id = NA, kappa = "2.0",
  gamma_site_model = create_gamma_site_model(),
  kappa_prior_distr = create_log_normal_distr(m = create_m_param(value =
  "1.0"), s = create_s_param(value = "1.25")), freq_equilibrium = "estimated")
```

Arguments

`id` the IDs of the alignment (can be extracted from the FASTA filesname using `get_id`)

`kappa` the kappa

`gamma_site_model` a gamma site model, as created by `create_gamma_site_model`

`kappa_prior_distr` the distribution of the kappa prior, which is a log-normal distribution (as created by `create_log_normal_distr`) by default

`freq_equilibrium` the frequency in which the rates are at equilibrium are either `estimated`, `empirical` or `all_equal`. `get_freq_equilibrium_names` returns the possible values for `freq_equilibrium`

Value

an HKY site_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
hky_site_model <- create_hky_site_model()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "beast.xml",
  site_models = hky_site_model
)
```

create_inv_gamma_distr *Create an inverse gamma distribution*

Description

Create an inverse gamma distribution

Usage

```
create_inv_gamma_distr(id = NA, alpha = create_alpha_param(),
  beta = create_beta_param())
```

Arguments

id	the distribution's ID
alpha	the alpha shape parameter, as returned by create_alpha_param
beta	the beta shape parameter, as returned by create_beta_param

Value

an inverse gamma distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
inv_gamma_distr <- create_inv_gamma_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "my_beast.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = inv_gamma_distr
  )
)
testit::assert(file.exists("my_beast.xml"))
```

---

create_jc69_site_model *Create a JC69 site model*

Description

Create a JC69 site model

Usage

```
create_jc69_site_model(id = NA,
  gamma_site_model = create_gamma_site_model())
```

Arguments

id	the IDs of the alignment (can be extracted from the FASTA filesname using get_id)
gamma_site_model	a gamma site model, as created by create_gamma_site_model

Value

a JC69 site_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
jc69_site_model <- create_jc69_site_model()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "beast.xml",
  site_models = jc69_site_model
)
```

`create_jc69_site_models` *Creates a JC69_site_model for each ID*

Description

Creates a JC69_site_model for each ID

Usage

```
create_jc69_site_models(ids)
```

Arguments

`ids`	one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with `get_ids`)

Value

a list of site_models

Author(s)

Richel J.C. Bilderbeek

See Also

The alignment IDs can be deduced from the FASTA filenames, using `get_ids`

Examples

```
fasta_filenames <- get_beautier_paths(
  c("anthus_aco.fas", "anthus_nd2.fas")
)
site_models <- create_jc69_site_models(c("anthus_aco", "anthus_nd2"))
create_beast2_input_file(
  fasta_filenames,
  "create_jc69_site_models.xml",
  site_models = site_models
)
testit::assert(file.exists("create_jc69_site_models.xml"))
```

`create_kappa_1_param` *Create a parameter called kappa 1*

Description

Create a parameter called kappa 1

Usage

```
create_kappa_1_param(id = NA, lower = "0.0", value = "2.0")
```

Arguments

`id`	the parameter's ID
`lower`	lowest possible value of the parameter
`value`	value of the parameter

Value

a parameter called kappa 1

Author(s)

Richel J.C. Bilderbeek

`create_kappa_2_param` *Create a parameter called kappa 2*

Description

Create a parameter called kappa 2

Usage

```
create_kappa_2_param(id = NA, lower = "0.0", value = "2.0")
```

Arguments

`id`	the parameter's ID
`lower`	lowest possible value of the parameter
`value`	value of the parameter

Value

a parameter called kappa 2

Author(s)

Richel J.C. Bilderbeek

---

create_lambda_param *Create a parameter called lambda*

Description

Create a parameter called lambda

Usage

```
create_lambda_param(id = NA, value = 0)
```

Arguments

id	the parameter's ID
value	value of the parameter

Value

a parameter called lambda

Note

this parameter is used in a Poisson distribution (as returned by create_poisson_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
lambda_param <- create_lambda_param()

# Use the parameter in a distribution
poisson_distr <- create_poisson_distr(
  lambda = lambda_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_lambda_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = poisson_distr
  )
)
testit::assert(file.exists("create_lambda_param.xml"))
```

create_laplace_distr *Create a Laplace distribution*

Description

Create a Laplace distribution

Usage

```
create_laplace_distr(id = NA, mu = create_mu_param(id = NA, estimate =
  FALSE, value = 0), scale = create_scale_param(id = NA, estimate = FALSE,
  value = 1))
```

Arguments

id	the distribution's ID
mu	the mu parameter, as returned by `create_mu_param`
scale	the scale parameter, as returned by `create_scale_param`

Value

a Laplace distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_distr` shows an overview of all supported distributions

Examples

```
laplace_distr <- create_laplace_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_laplace_distr.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = laplace_distr
  )
)
testit::assert(file.exists("create_laplace_distr.xml"))
```

create_log_normal_distr

Create a log-normal distribution

Description

Create a log-normal distribution

Usage

```
create_log_normal_distr(id = NA, m = create_m_param(),
  s = create_s_param())
```

Arguments

id	the distribution's ID
m	the m parameter, as returned by create_m_param
s	the s parameter, as returned by create_s_param

Value

a log-normal distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
log_normal_distr <- create_log_normal_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "my_beast.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = log_normal_distr
  )
)
testit::assert(file.exists("my_beast.xml"))
```

create_mcmc *Function to create the MCMC options, as in the BEAUti MCMC tab.*

Description

Function to create the MCMC options, as in the BEAUti MCMC tab.

Usage

```
create_mcmc(chain_length = 1e+07, store_every = -1)
```

Arguments

`chain_length`	the MCMC's chain length
`store_every`	number of states the posterior will be saved to file. Use -1 or NA to use the default frequency

Value

an mcmc

Author(s)

Richel J.C. Bilderbeek

Examples

```
mcmc <- create_mcmc(chain_length = 50000)

create_beast2_input_file(
  get_fasta_filename(),
  "create_mcmc.xml",
  mcmc = mcmc
)
testit::assert(file.exists("create_mcmc.xml"))
```

create_mean_param *Create a parameter called mean*

Description

Create a parameter called mean

Usage

```
create_mean_param(id = NA, estimate = FALSE, value = 0)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter alpha be estimated by BEAST2, FALSE otherwise
value	value of the parameter

Value

a parameter called mean

Note

this parameter is used in an exponential distribution (as returned by `create_exp_distr`) and normal distribution (as returned by `create_normal_distr`)

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create the parameter
mean_param <- create_mean_param(value = 1.0)

# Use the parameter in a distribution
exp_distr <- create_exp_distr(
  mean = mean_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_mean_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = exp_distr
  )
)
testit::assert(file.exists("create_mean_param.xml"))
```

create_misc_options *Function to create a misc_options, containing miscellaneous options to fine-tune the created BEAST2 XML file. Whatever option chosen here, the created XML file will be valid.*

Description

Function to create a misc_options, containing miscellaneous options to fine-tune the created BEAST2 XML file. Whatever option chosen here, the created XML file will be valid.

Usage

```
create_misc_options(capitalize_first_char_id = FALSE,
  nucleotides_uppercase = FALSE)
```

Arguments

`capitalize_first_char_id`
: must the ID of alignment start with a capital? TRUE if yes, FALSE if it can be left lower case (if it is lowercase)

`nucleotides_uppercase`
: must the nucleotides of the DNA sequence be in uppercase?

Value

a misc_options

Author(s)

Richel J.C. Bilderbeek

Examples

```
misc_options <- create_misc_options(nucleotides_uppercase = TRUE)
xml <- create_beast2_input(
  get_fasta_filename(),
  misc_options = misc_options
)
testit::assert(is.character(xml))
testit::assert(length(xml) > 1)
```

create_mrca_prior *Create a Most Recent Common Ancestor prior*

Description

Create a Most Recent Common Ancestor prior

Usage

```
create_mrca_prior(alignment_id, taxa_names, is_monophyletic = FALSE,
  mrca_distr = NA, name = NA, clock_prior_distr_id = NA)
```

Arguments

alignment_id	ID of the alignment, as returned by get_alignment_id
taxa_names	names of the taxa, as returned by get_taxa_names
is_monophyletic	boolean to indicate monophyly is assumed in a Most Recent Common Ancestor prior, as returned by create_mrca_prior
mrca_distr	the distribution used by the MRCA prior. Can be NA (the default) or any distribution returned by create_distr
name	the unique name of the MRCA prior, for example a genus, family, order or even class name. Leave at NA to have it named automatically
clock_prior_distr_id	ID of an MRCA clock model's distribution. Keep at NA to have it initialized automatically

Value

an MRCA prior

Author(s)

Richel J.C. Bilderbeek

create_mu_param *Create a parameter called mu*

Description

Create a parameter called mu

Usage

```
create_mu_param(id = NA, estimate = FALSE, value = 0)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
value	value of the parameter

Value

a parameter called mu

Note

this parameter is used in a Laplace distribution (as returned by create_laplace_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
mu_param <- create_mu_param()

# Use the parameter in a distribution
laplace_distr <- create_laplace_distr(
  mu = mu_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_mu_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = laplace_distr
  )
)
testit::assert(file.exists("create_mu_param.xml"))
```

create_m_param *Create a parameter called m*

Description

Create a parameter called m

Usage

```
create_m_param(id = NA, estimate = FALSE, value = 0)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter alpha be estimated by BEAST2, FALSE otherwise
value	value of the parameter

Value

a parameter called m

Note

this parameter is used in a log-normal distribution (as returned by create_log_normal_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
m_param <- create_m_param()

# Use the parameter in a distribution
log_normal_distr <- create_log_normal_distr(
  m = m_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_m_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = log_normal_distr
  )
)
testit::assert(file.exists("create_m_param.xml"))
```

create_normal_distr *Create an normal distribution*

Description

Create an normal distribution

Usage

```
create_normal_distr(id = NA, mean = create_mean_param(),
  sigma = create_sigma_param())
```

Arguments

id	the distribution's ID
mean	the mean parameter as returned by create_mean_param
sigma	the sigma parameter as returned by create_sigma_param

Value

a normal distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
normal_distr <- create_normal_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "my_beast.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = normal_distr
  )
)
testit::assert(file.exists("my_beast.xml"))
```

`create_one_div_x_distr` *Create a 1/x distribution*

Description

Create a 1/x distribution

Usage

```
create_one_div_x_distr(id = NA)
```

Arguments

id	the distribution's ID

Value

a 1/x distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
one_div_x_distr <- create_one_div_x_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "my_beast.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = one_div_x_distr
  )
)
testit::assert(file.exists("my_beast.xml"))
```

create_param *General function to create a parameter.*

Description

General function to create a parameter.

Usage

```
create_param(name, id, ...)
```

Arguments

name	the parameters' name. Valid names can be found in get_param_names
id	the parameter's ID
...	specific parameter parameters

Value

a parameter

Note

Prefer using the named functions create_alpha_param, create_beta_param, create_clock_rate_param, create_kappa_1_param, create_kappa_2_param, create_lambda_param, create_m_param, create_mean_param, create_mu_param, create_rate_ac_param, create_rate_ag_param, create_rate_at_param, create_rate_cg_param, create_rate_ct_param, create_rate_gt_param, create_s_param, create_scale_param, and create_sigma_param

Author(s)

Richel J.C. Bilderbeek

Examples

```
# Create an alpha parameter
alpha_param <- create_alpha_param()

# Use the parameter in a distribution
beta_distr <- create_beta_distr(
  alpha = alpha_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_alpha_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = beta_distr
  )
)
testit::assert(file.exists("create_alpha_param.xml"))
```

create_poisson_distr *Create a Poisson distribution*

Description

Create a Poisson distribution

Usage

```
create_poisson_distr(id = NA, lambda = create_lambda_param())
```

Arguments

id	the distribution's ID
lambda	the lambda parameter as returned by create_lambda_param

Value

a Poisson distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
poisson_distr <- create_poisson_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_poisson_distr.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = poisson_distr
  )
)
testit::assert(file.exists("create_poisson_distr.xml"))
```

create_random_name *Create a random name, for example, to be used by create_mrca_prior*

Description

Create a random name, for example, to be used by create_mrca_prior

Usage

```
create_random_name()
```

Value

a random name

Author(s)

Richel J.C. Bilderbeek

create_rate_ac_param *Create a parameter called 'rate AC'*

Description

Create a parameter called 'rate AC'

Usage

```
create_rate_ac_param(id = NA, estimate = TRUE, value = "1.0",
  lower = "0.0")
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
value	value of the parameter
lower	lowest possible value of the parameter

Value

a parameter called 'rate AC'

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create parameter
rate_ac_param <- create_rate_ac_param(value = 1, estimate = FALSE)

# Use the parameter to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_rate_ac_param.xml",
  site_models = create_gtr_site_model(
    rate_ac_param = rate_ac_param
  )
)
testit::assert(file.exists("create_rate_ac_param.xml"))
```

`create_rate_ag_param` *Create a parameter called 'rate AG'*

Description

Create a parameter called 'rate AG'

Usage

```
create_rate_ag_param(id = NA, estimate = TRUE, value = "1.0",
  lower = "0.0")
```

Arguments

`id`	the parameter's ID
`estimate`	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
`value`	value of the parameter
`lower`	lowest possible value of the parameter

Value

a parameter called 'rate AG'

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create parameter
rate_ag_param <- create_rate_ag_param(value = 1, estimate = FALSE)

# Use the parameter to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_rate_ag_param.xml",
  site_models = create_gtr_site_model(
    rate_ag_param = rate_ag_param
  )
)
testit::assert(file.exists("create_rate_ag_param.xml"))
```

`create_rate_at_param` *Create a parameter called 'rate AT'*

Description

Create a parameter called 'rate AT'

Usage

```
create_rate_at_param(id = NA, estimate = TRUE, value = "1.0",
  lower = "0.0")
```

Arguments

`id`	the parameter's ID
`estimate`	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
`value`	value of the parameter
`lower`	lowest possible value of the parameter

Value

a parameter called 'rate AT'

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create parameter
rate_at_param <- create_rate_at_param(value = 1, estimate = FALSE)

# Use the parameter to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_rate_at_param.xml",
  site_models = create_gtr_site_model(
    rate_at_param = rate_at_param
  )
)
testit::assert(file.exists("create_rate_at_param.xml"))
```

`create_rate_cg_param` *Create a parameter called 'rate CG'*

Description

Create a parameter called 'rate CG'

Usage

```
create_rate_cg_param(id = NA, estimate = TRUE, value = "1.0",
  lower = "0.0")
```

Arguments

`id`	the parameter's ID
`estimate`	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
`value`	value of the parameter
`lower`	lowest possible value of the parameter

Value

a parameter called 'rate CG'

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create parameter
rate_cg_param <- create_rate_cg_param(value = 1, estimate = FALSE)

# Use the parameter to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_rate_cg_param.xml",
  site_models = create_gtr_site_model(
    rate_cg_param = rate_cg_param
  )
)
testit::assert(file.exists("create_rate_cg_param.xml"))
```

---

create_rate_ct_param *Create a parameter called 'rate CT'*

Description

Create a parameter called 'rate CT'

Usage

```
create_rate_ct_param(id = NA, estimate = TRUE, value = "1.0",
  lower = "0.0")
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
value	value of the parameter
lower	lowest possible value of the parameter

Value

a parameter called 'rate CT'

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create parameter
rate_ct_param <- create_rate_ct_param(value = 1, estimate = FALSE)

# Use the parameter to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_rate_ct_param.xml",
  site_models = create_gtr_site_model(
    rate_ct_param = rate_ct_param
  )
)
testit::assert(file.exists("create_rate_ct_param.xml"))
```

`create_rate_gt_param` *Create a parameter called 'rate GT'*

Description

Create a parameter called 'rate GT'

Usage

```
create_rate_gt_param(id = NA, estimate = TRUE, value = "1.0",
  lower = "0.0")
```

Arguments

`id`	the parameter's ID
`estimate`	TRUE if this parameter mu be estimated by BEAST2, FALSE otherwise
`value`	value of the parameter
`lower`	lowest possible value of the parameter

Value

a parameter called 'rate GT'

Author(s)

Richel J.C. Bilderbeek

See Also

the function `create_param` contains a list of all parameters that can be created

Examples

```
# Create parameter
rate_gt_param <- create_rate_gt_param(value = 1, estimate = FALSE)

# Use the parameter to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_rate_gt_param.xml",
  site_models = create_gtr_site_model(
    rate_gt_param = rate_gt_param
  )
)
testit::assert(file.exists("create_rate_gt_param.xml"))
```

---

create_rln_clock_model

Create a relaxed log-normal clock model

Description

Create a relaxed log-normal clock model

Usage

```
create_rln_clock_model(id = NA,
  mean_rate_prior_distr = create_uniform_distr(),
  ucldstdev_distr = create_gamma_distr(), mparam_id = NA,
  mean_clock_rate = "1.0", n_rate_categories = -1,
  normalize_mean_clock_rate = FALSE, dimension = NA)
```

Arguments

id	an alignment's IDs. An ID can be extracted from its FASTA filesname with get_ids)
mean_rate_prior_distr	the mean clock rate prior distribution, as created by a create_distr function
ucldstdev_distr	the uclstdev distribution, as created by a create_distr function
mparam_id	the ID of the M paramater in the branchRateModel, set to NA to have it initialized
mean_clock_rate	the mean clock rate, 1.0 by default (is called 'ucld_stdev' in XML, where 'ucld_stdev' is always 0.1)
n_rate_categories	the number of rate categories. -1 is default, 0 denotes as much rates as branches
normalize_mean_clock_rate	normalize the mean clock rate

`dimension` the dimensionality of the relaxed clock model. Leave NA to let beautier calculate it. Else, the dimensionality of the clock equals twice the number of taxa minus two.

Value

a relaxed log-normal clock_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
rln_clock_model <- create_rln_clock_model()

create_beast2_input_file(
  get_fasta_filename(),
  "create_rln_clock_model.xml",
  clock_models = rln_clock_model
)
testit::assert(file.exists("create_rln_clock_model.xml"))

rln_clock_model_exp <- create_rln_clock_model(
  mean_rate_prior_distr = create_exp_distr()
)

create_beast2_input_file(
  get_fasta_filename(),
  "create_rln_clock_model_exp.xml",
  clock_models = rln_clock_model_exp
)
testit::assert(file.exists("create_rln_clock_model_exp.xml"))
```

---

create_rnd_alpha_param

Create a random alpha parameter

Description

Create a random alpha parameter

Usage

```
create_rnd_alpha_param()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_bd_tree_prior *Create a random BD tree prior*

Description

Create a random BD tree prior

Usage

```
create_rnd_bd_tree_prior()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_beta_distr *Create a random beta distribution*

Description

Create a random beta distribution

Usage

```
create_rnd_beta_distr()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_beta_param *Create a random beta parameter*

Description

Create a random beta parameter

Usage

```
create_rnd_beta_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_bool` *Create a random boolean*

Description

Create a random boolean

Usage

```
create_rnd_bool()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_cbs_tree_prior` *Create a random CBS tree prior*

Description

Create a random CBS tree prior

Usage

```
create_rnd_cbs_tree_prior()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_ccp_tree_prior` *Create a random CCP tree prior*

Description

Create a random CCP tree prior

Usage

```
create_rnd_ccp_tree_prior()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_cep_tree_prior

Create a random CEP tree prior

Description

Create a random CEP tree prior

Usage

```
create_rnd_cep_tree_prior()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_clock_model

Create a random clock model

Description

Create a random clock model

Usage

```
create_rnd_clock_model()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_clock_rate_param

Create a random clock rate parameter

Description

Create a random clock rate parameter

Usage

```
create_rnd_clock_rate_param()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_distr *Create a random distribution*

Description

Create a random distribution

Usage

```
create_rnd_distr()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_estimate *Create a random value for 'estimate', which must be TRUE or FALSE*

Description

Create a random value for 'estimate', which must be TRUE or FALSE

Usage

```
create_rnd_estimate()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_exp_distr *Create a random exponential distribution*

Description

Create a random exponential distribution

Usage

```
create_rnd_exp_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_freq_equilibrium`
Create a random value for 'freq equilibrium', which can be 'estimated', 'empirical' or 'all_equal'

Description

Create a random value for 'freq equilibrium', which can be 'estimated', 'empirical' or 'all_equal'

Usage

```
create_rnd_freq_equilibrium()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_gamma_distr`
Create a random gamma distribution

Description

Create a random gamma distribution

Usage

```
create_rnd_gamma_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_gamma_site_model` *Create a random gamma site model*

Description

Create a random gamma site model

Usage

```
create_rnd_gamma_site_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_gtr_site_model` *Create a random GTR site model*

Description

Create a random GTR site model

Usage

```
create_rnd_gtr_site_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_hky_site_model` *Create a random HKY site model*

Description

Create a random HKY site model

Usage

```
create_rnd_hky_site_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_inv_gamma_distr` *Create a random inverse-gamma distribution*

Description

Create a random inverse-gamma distribution

Usage

```
create_rnd_inv_gamma_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_jc69_site_model` *Create a random JC69 distribution*

Description

Create a random JC69 distribution

Usage

```
create_rnd_jc69_site_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_kappa_1_param` *Create a random kappa 1 parameter*

Description

Create a random kappa 1 parameter

Usage

```
create_rnd_kappa_1_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_kappa_2_param` *Create a random kappa 2 parameter*

Description

Create a random kappa 2 parameter

Usage

```
create_rnd_kappa_2_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_lambda_param` *Create a random lambda parameter*

Description

Create a random lambda parameter

Usage

```
create_rnd_lambda_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_laplace_distr` *Create a random Laplace distribution*

Description

Create a random Laplace distribution

Usage

```
create_rnd_laplace_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_log_normal_distr` *Create a random log-normal distribution*

Description

Create a random log-normal distribution

Usage

```
create_rnd_log_normal_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_mean_param` *Create a random mean parameter*

Description

Create a random mean parameter

Usage

```
create_rnd_mean_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_mrca_prior` *Create a random MRCA prior*

Description

Create a random MRCA prior

Usage

```
create_rnd_mrca_prior(fasta_filename)
```

Arguments

`fasta_filename`	a FASTA filename

Author(s)

Richel J.C. Bilderbeek

`create_rnd_mrca_priors` *Creates a random MRCA prior option. This is either zero, one or more MRCA priors. For zero MRCA priors, NA is used. When two or more MRCA priors (see create_rnd_two_mrca_priors) are created, those are checked to be compatible*

Description

Creates a random MRCA prior option. This is either zero, one or more MRCA priors. For zero MRCA priors, NA is used. When two or more MRCA priors (see create_rnd_two_mrca_priors) are created, those are checked to be compatible

Usage

```
create_rnd_mrca_priors(fasta_filename)
```

Arguments

`fasta_filename`	a FASTA filename

Author(s)

Richel J.C. Bilderbeek

`create_rnd_mu_param` *Create a random mu parameter*

Description

Create a random mu parameter

Usage

```
create_rnd_mu_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_m_param` *Create a random m parameter*

Description

Create a random m parameter

Usage

```
create_rnd_m_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_normal_distr` *Create a random normal distribution*

Description

Create a random normal distribution

Usage

```
create_rnd_normal_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_one_div_x_distr` *Create a random 1/x distribution*

Description

Create a random 1/x distribution

Usage

```
create_rnd_one_div_x_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_param` *Create a random parameter*

Description

Create a random parameter

Usage

```
create_rnd_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_poisson_distr` *Create a random Poisson distribution*

Description

Create a random Poisson distribution

Usage

```
create_rnd_poisson_distr()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rate_ac_param` *Create a random rate AC parameter*

Description

Create a random rate AC parameter

Usage

```
create_rnd_rate_ac_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rate_ag_param`

Create a random rate AG parameter

Description

Create a random rate AG parameter

Usage

```
create_rnd_rate_ag_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rate_at_param`

Create a random rate AT parameter

Description

Create a random rate AT parameter

Usage

```
create_rnd_rate_at_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rate_cg_param`

Create a random rate CG parameter

Description

Create a random rate CG parameter

Usage

```
create_rnd_rate_cg_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rate_ct_param` *Create a random rate CT parameter*

Description

Create a random rate CT parameter

Usage

```
create_rnd_rate_ct_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rate_gt_param` *Create a random rate GT parameter*

Description

Create a random rate GT parameter

Usage

```
create_rnd_rate_gt_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_rln_clock_model` *Create a random RLN clock model*

Description

Create a random RLN clock model

Usage

```
create_rnd_rln_clock_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_scale_param` *Create a random scale parameter*

Description

Create a random scale parameter

Usage

```
create_rnd_scale_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_sigma_param` *Create a random sigma parameter*

Description

Create a random sigma parameter

Usage

```
create_rnd_sigma_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_site_model` *Create a random site model*

Description

Create a random site model

Usage

```
create_rnd_site_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_strict_clock_model` *Create a random strict clock model*

Description

Create a random strict clock model

Usage

```
create_rnd_strict_clock_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_s_param` *Create a random s parameter*

Description

Create a random s parameter

Usage

```
create_rnd_s_param()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_tn93_site_model` *Create a random TN93 site model*

Description

Create a random TN93 site model

Usage

```
create_rnd_tn93_site_model()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_tree_prior` *Create a random tree prior*

Description

Create a random tree prior

Usage

```
create_rnd_tree_prior()
```

Author(s)

Richel J.C. Bilderbeek

`create_rnd_two_mrca_priors` *Creates two MRCA priors, checked to be compatible*

Description

Creates two MRCA priors, checked to be compatible

Usage

```
create_rnd_two_mrca_priors(fasta_filename)
```

Arguments

`fasta_filename`	a FASTA filename

Author(s)

Richel J.C. Bilderbeek

create_rnd_uniform_distr

Create a random uniform distribution

Description

Create a random uniform distribution

Usage

```
create_rnd_uniform_distr()
```

Author(s)

Richel J.C. Bilderbeek

create_rnd_yule_tree_prior

Create a random Yule tree prior

Description

Create a random Yule tree prior

Usage

```
create_rnd_yule_tree_prior()
```

Author(s)

Richel J.C. Bilderbeek

create_scale_param *Create a parameter called scale*

Description

Create a parameter called scale

Usage

```
create_scale_param(id = NA, estimate = FALSE, value = 0)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter scale be estimated by BEAST2, FALSE otherwise
value	value of the parameter

Value

a parameter called scale

Note

this parameter is used in a Laplace distribution (as returned by create_laplace_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
scale_param <- create_scale_param()

# Use the parameter in a distribution
laplace_distr <- create_laplace_distr(
  scale = scale_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_scale_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = laplace_distr
  )
)
testit::assert(file.exists("create_scale_param.xml"))
```

create_sigma_param *Create a parameter called sigma*

Description

Create a parameter called sigma

Usage

```
create_sigma_param(id = NA, estimate = FALSE, value = 1)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter alpha be estimated by BEAST2, FALSE otherwise
value	value of the parameter

Value

a parameter called sigma

Note

this parameter is used in a normal distribution (as returned by create_normal_distr)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
sigma_param <- create_sigma_param()

# Use the parameter in a distribution
normal_distr <- create_normal_distr(
  sigma = sigma_param
)

# Use the distribution to create a BEAST2 input file
```

```
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_sigma_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = normal_distr
  )
)
testit::assert(file.exists("create_sigma_param.xml"))
```

create_site_model *General function to create a site model.*

Description

General function to create a site model.

Usage

```
create_site_model(name, id, gamma_site_model = create_gamma_site_model(), ...)
```

Arguments

`name`	the site model name. Valid names can be found in `get_site_model_names`
`id`	the IDs of the alignment (can be extracted from the FASTA filesname using `get_id`)
`gamma_site_model`	a gamma site model, as created by `create_gamma_site_model`
`...`	specific site model parameters

Value

a site_model

Note

Prefer using the named functions `create_gtr_site_model`, `create_hky_site_model`,, `create_jc69_site_model`, and `create_tn93_site_model`

Author(s)

Richel J.C. Bilderbeek

See Also

See `create_gtr_site_model` for more examples with a GTR site model. See `create_hky_site_model` for more examples with an HKY site model. See `create_jc69_site_model` for more examples with a JC69 site model. See `create_tn93_site_model` for more examples with a TN93 site model

Examples

```
# GTR
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  output_filename = "example_gtr.xml",
  site_models = create_gtr_site_model()
)
testthat::expect_true(file.exists("example_gtr.xml"))

# HKY
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  output_filename = "example_hky.xml",
  site_models = create_hky_site_model()
)
testthat::expect_true(file.exists("example_hky.xml"))

# JC69
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  output_filename = "example_jc69.xml",
  site_models = create_jc69_site_model()
)
testthat::expect_true(file.exists("example_jc69.xml"))

# TN93
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  output_filename = "example_tn93.xml",
  site_models = create_tn93_site_model()
)
testthat::expect_true(file.exists("example_tn93.xml"))
```

`create_site_models` *Creates all supported site models which is just a list of the types returned by `create_gtr_site_model`, `create_hky_site_model`, `create_jc69_site_model` and `create_tn93_site_model`*

Description

Creates all supported site models which is just a list of the types returned by `create_gtr_site_model`, `create_hky_site_model`, `create_jc69_site_model` and `create_tn93_site_model`

Usage

```
create_site_models()
```

Value

a list of site_models

Author(s)

Richel J.C. Bilderbeek

Examples

```
site_models <- beautier:::create_site_models()
testit::assert(beautier:::is_gtr_site_model(site_models[[1]]))
testit::assert(beautier:::is_hky_site_model(site_models[[2]]))
testit::assert(beautier:::is_jc69_site_model(site_models[[3]]))
testit::assert(beautier:::is_tn93_site_model(site_models[[4]]))
```

---

create_strict_clock_model
Create a strict clock model

Description

Create a strict clock model

Usage

```
create_strict_clock_model(id = NA,
  clock_rate_param = create_clock_rate_param(),
  clock_rate_distr = create_uniform_distr())
```

Arguments

`id` an alignment's IDs. An ID can be extracted from its FASTA filesname with `get_ids`)

`clock_rate_param` the clock rate's parameter, as created by the `create_clock_rate_param` function

`clock_rate_distr` the clock rate's distribution, as created by a `create_distr` function

Value

a strict clock_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
strict_clock_model <- create_strict_clock_model()

create_beast2_input_file(
  get_fasta_filename(),
  "create_strict_clock_model.xml",
  clock_models = strict_clock_model
)
testit::assert(file.exists("create_strict_clock_model.xml"))

strict_clock_model_gamma <- create_strict_clock_model(
  clock_rate_distr = create_gamma_distr()
)

create_beast2_input_file(
  get_fasta_filename(),
  "create_strict_clock_model_gamma.xml",
  clock_models = strict_clock_model_gamma
)
testit::assert(file.exists("create_strict_clock_model_gamma.xml"))
```

---

create_strict_clock_models *Creates n strict clock_models*

Description

Creates n strict clock_models

Usage

```
create_strict_clock_models(ids)
```

Arguments

ids	the alignment IDs

Value

a list of strict_clock objects

See Also

The alignment IDs can be deduced from the FASTA filenames, using get_ids

Examples

```
fasta_filenames <- get_beautier_paths(
  c("anthus_aco.fas", "anthus_nd2.fas")
)
clock_models <- create_strict_clock_models(
  ids = get_ids(fasta_filenames)
)

create_beast2_input_file(
  fasta_filenames,
  "create_strict_clock_models.xml",
  clock_models = clock_models
)
testit::assert(file.exists("create_strict_clock_models.xml"))
```

create_s_param *Create a parameter called s*

Description

Create a parameter called s

Usage

```
create_s_param(id = NA, estimate = FALSE, value = 0, lower = 0,
  upper = 0)
```

Arguments

id	the parameter's ID
estimate	TRUE if this parameter alpha be estimated by BEAST2, FALSE otherwise
value	value of the parameter
lower	lower value of the parameter
upper	upper value of the parameter

Value

a parameter called s

Note

this parameter is used in a log-normal distribution (as returned by `create_log_normal_distr`)

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_param contains a list of all parameters that can be created

Examples

```
# Create the parameter
s_param <- create_s_param()

# Use the parameter in a distribution
log_normal_distr <- create_log_normal_distr(
  s = s_param
)

# Use the distribution to create a BEAST2 input file
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_s_param.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = log_normal_distr
  )
)
testit::assert(file.exists("create_s_param.xml"))
```

---

create_tn93_site_model

Create a TN93 site model

Description

Create a TN93 site model

Usage

```
create_tn93_site_model(id = NA,
  gamma_site_model = create_gamma_site_model(),
  kappa_1_param = create_kappa_1_param(),
  kappa_2_param = create_kappa_2_param(),
  kappa_1_prior_distr = create_log_normal_distr(m = create_m_param(id = NA,
  estimate = FALSE, value = "1.0"), s = create_s_param(id = NA, estimate =
  FALSE, value = "1.25")), kappa_2_prior_distr = create_log_normal_distr(m =
  create_m_param(id = NA, estimate = FALSE, value = "1.0"), s =
  create_s_param(id = NA, estimate = FALSE, value = "1.25")),
  freq_equilibrium = "estimated")
```

Arguments

id	the IDs of the alignment (can be extracted from the FASTA filesname using get_id)

`gamma_site_model`
a gamma site model, as created by `create_gamma_site_model`

`kappa_1_param` the ’kappa 1’ parameter, as returned by `create_kappa_1_param`

`kappa_2_param` the ’kappa 2’ parameter, as returned by `create_kappa_2_param`

`kappa_1_prior_distr`
the distribution of the kappa 1 prior, which is a log-normal distribution (as created by `create_log_normal_distr`) by default

`kappa_2_prior_distr`
the distribution of the kappa 2 prior, which is a log-normal distribution (as created by `create_log_normal_distr`) by default

`freq_equilibrium`
the frequency in which the rates are at equilibrium are either `estimated`, `empirical` or `all_equal`. `get_freq_equilibrium_names` returns the possible values for `freq_equilibrium`

Value

a TN93 site_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
tn93_site_model <- create_tn93_site_model()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "beast.xml",
  site_models = tn93_site_model
)
```

`create_tree_prior` *Internal function to create a tree prior*

Description

Internal function to create a tree prior

Usage

```
create_tree_prior(name, id, ...)
```

Arguments

name	the tree prior name. Can be any name in get_tree_prior_names
id	the ID of the alignment
...	specific tree prior parameters

Value

a tree_prior

Note

Prefer the use the named functions create_bd_tree_prior, create_cbs_tree_prior, create_ccp_tree_prior create_cep_tree_prior and create_yule_tree_prior instead

Author(s)

Richel J.C. Bilderbeek

See Also

See create_bd_tree_prior, create_cbs_tree_prior, create_ccp_tree_prior create_cep_tree_prior and create_yule_tree_prior for more examples using those functions

Examples

```
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_tree_prior_bd.xml",
  tree_priors = create_bd_tree_prior()
)
testit::assert(file.exists("create_tree_prior_bd.xml"))

create_beast2_input_file(
  input_filenames = get_beautier_path("test_output_6.fas"),
  "create_tree_prior_cbs.xml",
  tree_priors = create_cbs_tree_prior()
)
testit::assert(file.exists("create_tree_prior_cbs.xml"))

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_tree_prior_ccp.xml",
  tree_priors = create_ccp_tree_prior()
)
testit::assert(file.exists("create_tree_prior_ccp.xml"))

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_tree_prior_cep.xml",
  tree_priors = create_cep_tree_prior()
)
```

```
testit::assert(file.exists("create_tree_prior_cep.xml"))

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_tree_prior_yule.xml",
  tree_priors = create_yule_tree_prior()
)
testit::assert(file.exists("create_tree_prior_yule.xml"))
```

`create_tree_priors` *Creates all supported tree priors, which is just a list of the types returned by* `create_bd_tree_prior`*,* `create_cbs_tree_prior`*,* `create_ccp_tree_prior`*,* `create_cep_tree_prior` *and* `create_yule_tree_prior`

Description

Creates all supported tree priors, which is just a list of the types returned by `create_bd_tree_prior`, `create_cbs_tree_prior`, `create_ccp_tree_prior`, `create_cep_tree_prior` and `create_yule_tree_prior`

Usage

```
create_tree_priors()
```

Value

a list of tree_priors

Author(s)

Richel J.C. Bilderbeek

Examples

```
tree_priors <- beautier:::create_tree_priors()
testit::assert(beautier:::is_bd_tree_prior(tree_priors[[1]]))
testit::assert(beautier:::is_cbs_tree_prior(tree_priors[[2]]))
testit::assert(beautier:::is_ccp_tree_prior(tree_priors[[3]]))
testit::assert(beautier:::is_cep_tree_prior(tree_priors[[4]]))
testit::assert(beautier:::is_yule_tree_prior(tree_priors[[5]]))
```

create_uniform_distr *Create a uniform distribution*

Description

Create a uniform distribution

Usage

```
create_uniform_distr(id = NA, upper = Inf)
```

Arguments

id	the distribution's ID
upper	an upper limit of the uniform distribution. If the upper limits needs to be infinity, set upper to Inf.

Value

a uniform distribution

Author(s)

Richel J.C. Bilderbeek

See Also

the function create_distr shows an overview of all supported distributions

Examples

```
uniform_distr <- create_uniform_distr()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_uniform_distr.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = uniform_distr
  )
)
testit::assert(file.exists("create_uniform_distr.xml"))
```

create_yule_tree_prior

Create a Yule tree prior

Description

Create a Yule tree prior

Usage

```
create_yule_tree_prior(
  id = NA,
  birth_rate_distr = create_uniform_distr()
)
```

Arguments

`id`	the ID of the alignment
`birth_rate_distr`	the birth rate distribution, as created by a `create_distr` function

Value

a Yule tree_prior

Author(s)

Richel J.C. Bilderbeek

See Also

An alignment ID can be extracted from its FASTA filesname using `get_id`

Examples

```
yule_tree_prior <- create_yule_tree_prior()

create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "create_yule_tree_prior.xml",
  tree_priors = yule_tree_prior
)
testit::assert(file.exists("create_yule_tree_prior.xml"))
```

create_yule_tree_priors *Creates n Yule tree priors*

Description

Creates n Yule tree priors

Usage

```
create_yule_tree_priors(ids)
```

Arguments

`ids`	the alignment IDs

Value

a list of Yule tree_prior objects

Examples

```
fasta_filenames <- get_beautier_paths(
  c("anthus_aco.fas", "anthus_nd2.fas")
)

tree_priors <- create_yule_tree_priors(ids = get_ids(fasta_filenames))

create_beast2_input_file(
  fasta_filenames,
  "create_yule_tree_priors.xml",
  tree_priors = tree_priors
)
testit::assert(file.exists("create_yule_tree_priors.xml"))
```

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

Description

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

Usage

```
default_params_doc(alignment_id, bd_tree_prior, cbs_tree_prior, ccp_tree_prior,
  cep_tree_prior, clock_model, clock_models, clock_prior_distr_id, crown_age,
  crown_ages, distr_id, fasta_filename, fasta_filenames, fixed_crown_age,
  fixed_crown_ages, gamma_site_model, group_sizes_dimension, gtr_site_model,
  has_non_strict_clock_model, hky_site_model, id, ids, initial_phylogenies,
  input_filenames, is_first, is_monophyletic, jc69_site_model, mcmc,
  misc_options, mrca_prior, mrca_priors, output_filename, param_id,
  posterior_crown_age, rln_clock_model, sequence_length, site_model,
  site_models, strict_clock_model, tn93_site_model, tree_prior, tree_priors,
  verbose, yule_tree_prior)
```

Arguments

`alignment_id` ID of the alignment, as returned by `get_alignment_id`

`bd_tree_prior` a Birth-Death tree prior, as created by `create_bd_tree_prior`

`cbs_tree_prior` a Coalescent Bayesian Skyline tree prior, as returned by `create_cbs_tree_prior`

`ccp_tree_prior` a Coalescent Constant Population tree prior, as returned by `create_ccp_tree_prior`

`cep_tree_prior` a cep tree prior, as returned by `create_cep_tree_prior`

`clock_model` a clock model, as returned by `create_clock_model`

`clock_models` a list of one or more clock models, as returned by `create_clock_model`

`clock_prior_distr_id` ID of an MRCA clock model's distribution. Keep at NA to have it initialized automatically

`crown_age` the crown age of the phylogeny

`crown_ages` the crown ages of the phylogenies. Set to NA if the crown age needs to be estimated

`distr_id` a distributions' ID

`fasta_filename` a FASTA filename. Use `get_fasta_filename` to obtain a testing FASTA filename.

`fasta_filenames` One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`fixed_crown_age` determines if the phylogeny's crown age is fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, 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 FALSE, crown age is estimated by BEAST2. If TRUE, the crown age is fixed to the crown age of the initial phylogeny.

`gamma_site_model` a site model's gamma site model, as returned by `create_gamma_site_model`

`group_sizes_dimension` the group sizes' dimension, as used by the CBS tree prior (see `create_cbs_tree_prior`) `gtr_site_model` a GTR site model, as returned by `create_gtr_site_model`

`has_non_strict_clock_model`
boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

`hky_site_model` an HKY site model, as returned by `create_hky_site_model`

`id` an alignment's IDs. An ID can be extracted from its FASTA filesname with `get_ids`)

`ids` one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with `get_ids`)

`initial_phylogenies`
one or more MCMC chain's initial phylogenies. Each one set to NA will result in BEAST2 using a random phylogeny. Else the phylogeny is assumed to be of class ape::phylo.

`input_filenames`
One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.

`is_first` boolean to indicate if this is the first out of many (one, two or more)

`is_monophyletic`
boolean to indicate monophyly is assumed in a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`

`jc69_site_model`
a JC69 site model, as returned by `create_jc69_site_model`

`mcmc` one mcmc object, as returned by `create_mcmc`

`misc_options` one misc_options object, as returned by `create_misc_options`

`mrca_prior` a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`

`mrca_priors` a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

`output_filename`
Name of the XML parameter file created by this function. BEAST2 uses this file as input.

`param_id` a parameter's ID

`posterior_crown_age`
the crown age the posteriors' phylogenies will be fixed at. If NA, crown age is estimated by BEAST2.

`rln_clock_model`
a Relaxed Log-Normal clock model, as returned by `create_rln_clock_model`

`sequence_length`
a DNA sequence length, in base pairs

`site_model` a site model, as returned by `create_site_model`

`site_models` one or more site models, as returned by `create_site_model`

`strict_clock_model`
a strict clock model, as returned by `create_strict_clock_model`

`tn93_site_model`	a TN93 site model, as returned by `create_tn93_site_model`
`tree_prior`	a tree priors, as returned by `create_tree_prior`
`tree_priors`	one or more tree priors, as returned by `create_tree_prior`
`verbose`	if TRUE, additional information is displayed, that is potentially useful in debugging
`yule_tree_prior`	a Yule tree_prior, as created by `create_yule_tree_prior`

Author(s)

Richel J.C. Bilderbeek

`distr_to_xml` *Converts a distribution to XML*

Description

Converts a distribution to XML

Usage

```
distr_to_xml(distr)
```

Arguments

`distr`	a distibution, as created by `create_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

Examples

```
xml <- beautier:::distr_to_xml(create_uniform_distr(id = 1))
testit::assert(is.character(xml))
testit::assert(length(xml) == 1)
testit::assert(nchar(xml) > 1)
```

`distr_to_xml_beta` *Converts a beta distribution to XML*

Description

Converts a beta distribution to XML

Usage

```
distr_to_xml_beta(distr)
```

Arguments

`distr`	a beta distibution, as created by `create_beta_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

`distr_to_xml_exp` *Converts an exponential distribution to XML*

Description

Converts an exponential distribution to XML

Usage

```
distr_to_xml_exp(distr)
```

Arguments

`distr`	an exponential distibution, as created by `create_exp_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

`distr_to_xml_gamma` *Converts a gamma distribution to XML*

Description

Converts a gamma distribution to XML

Usage

```
distr_to_xml_gamma(distr)
```

Arguments

`distr`	a gamma distibution, as created by `create_gamma_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

`distr_to_xml_inv_gamma` *Converts a inv_gamma distribution to XML*

Description

Converts a inv_gamma distribution to XML

Usage

```
distr_to_xml_inv_gamma(distr)
```

Arguments

`distr`	a inv_gamma distibution, as created by `create_inv_gamma_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

distr_to_xml_laplace *Converts a laplace distibution to XML*

Description

Converts a laplace distibution to XML

Usage

```
distr_to_xml_laplace(distr)
```

Arguments

distr	a laplace distibution, as created by create_laplace_distr)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

distr_to_xml_log_normal *Converts a log-normal distribution to XML*

Description

Converts a log-normal distribution to XML

Usage

```
distr_to_xml_log_normal(distr)
```

Arguments

distr	a log-normal distibution, as created by create_log_normal_distr)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

`distr_to_xml_normal` *Converts a normal distribution to XML*

Description

Converts a normal distribution to XML

Usage

```
distr_to_xml_normal(distr)
```

Arguments

`distr`	a normal distibution, as created by `create_normal_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

`distr_to_xml_one_div_x` *Converts a 1/x distribution to XML*

Description

Converts a 1/x distribution to XML

Usage

```
distr_to_xml_one_div_x(distr)
```

Arguments

`distr`	a 1/x distibution, as created by `create_one_div_x_distr`)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

distr_to_xml_poisson *Converts a poisson distribution to XML*

Description

Converts a poisson distribution to XML

Usage

```
distr_to_xml_poisson(distr)
```

Arguments

distr	a poisson distibution, as created by create_poisson_distr)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

distr_to_xml_uniform *Converts a uniform distribution to XML*

Description

Converts a uniform distribution to XML

Usage

```
distr_to_xml_uniform(distr)
```

Arguments

distr	a uniform distibution, as created by create_uniform_distr)

Value

the distribution as XML text

Author(s)

Richel J.C. Bilderbeek

`extract_xml_loggers_from_lines` *Extract everything between first loggers and last loggers line*

Description

Extract everything between first loggers and last loggers line

Usage

```
extract_xml_loggers_from_lines(lines)
```

Arguments

`lines`	lines of text

Value

lines of text from the first to and including the last operators line

Author(s)

Richel J.C. Bilderbeek

`extract_xml_operators_from_lines` *Extract everything between first operators and last operators line*

Description

Extract everything between first operators and last operators line

Usage

```
extract_xml_operators_from_lines(lines)
```

Arguments

`lines`	lines of text

Value

lines of text from the first to and including the last operators line

Author(s)

Richel J.C. Bilderbeek

extract_xml_section_from_lines

Get the lines of an XML section, including the section tags

Description

Get the lines of an XML section, including the section tags

Usage

```
extract_xml_section_from_lines(lines, section)
```

Arguments

`lines`	lines of the XML text
`section`	the XML section name

Value

the section's lines of XML text, including the tags

Author(s)

Richel J.C. Bilderbeek

fastas_to_phylos

Create a random phylogeny, with the same taxa names as the FASTA file and the desired crown age

Description

Create a random phylogeny, with the same taxa names as the FASTA file and the desired crown age

Usage

```
fastas_to_phylos(fasta_filenames, crown_age)
```

Arguments

`fasta_filenames`	One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.
`crown_age`	the crown age of the phylogeny

Value

a a random phylogy, with the same taxa names as the FASTA file and the desired crown age

Examples

```
# Create two random phylogies, with
# - the same taxa names as the FASTA files
# - the desired crown age
fasta_filenames <- get_beautier_paths(
  c("anthus_aco.fas", "anthus_nd2.fas")
)
initial_phylogenies <- fastas_to_phylos(
  fasta_filenames,
  crown_age = 15
 )

# Crown age fixed to the crown age of the phylogeny
create_beast2_input_file_1_12(
  input_filenames = fasta_filenames,
  "fastas_to_phylos.xml",
  fixed_crown_ages = c(TRUE, TRUE),
  initial_phylogenies = initial_phylogenies
)
testthat::expect_true(file.exists("fastas_to_phylos.xml"))
```

---

fasta_file_to_sequences

Convert a FASTA file to a table of sequences

Description

Convert a FASTA file to a table of sequences

Usage

```
fasta_file_to_sequences(fasta_filenames)
```

Arguments

`fasta_filenames`
One or more existing FASTA filenames

Value

a table of sequences

Author(s)

Richel J.C. Bilderbeek

fasta_to_phylo *Create a random phylogeny, with the same taxa names as the FASTA file and the desired crown age*

Description

Create a random phylogeny, with the same taxa names as the FASTA file and the desired crown age

Usage

```
fasta_to_phylo(fasta_filename, crown_age)
```

Arguments

fasta_filename	a FASTA filename. Use get_fasta_filename to obtain a testing FASTA filename.
crown_age	the crown age of the phylogeny

Value

a a random phylogy, with the same taxa names as the FASTA file and the desired crown age

Examples

```
# Create a random phylogy, with
# - the same taxa names as the FASTA file
# - the desired crown age
fasta_filename <- get_fasta_filename()
initial_phylogeny <- fasta_to_phylo(
  fasta_filename,
  crown_age = 15
 )

# Crown age fixed to the crown age of the phylogeny
# Note: prefer create_beast2_input_file
create_beast2_input_file_1_12(
  input_filenames = fasta_filename,
  "fasta_to_phylo.xml",
  fixed_crown_age = TRUE,
  initial_phylogenies = initial_phylogeny
)
testthat::expect_true(file.exists("fasta_to_phylo.xml"))
```

files_exist *Multi-file file.exists function*

Description

Multi-file file.exists function

Usage

```
files_exist(filenames)
```

Arguments

`filenames`	one or more filename

Value

TRUE if all files exists, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

find_clock_model *Finds a clock model with a certain ID*

Description

Finds a clock model with a certain ID

Usage

```
find_clock_model(clock_models, id)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`
`id`	the ID of the clock model

Value

the clock models with the desired ID, NULL if such a clock model is absent

Author(s)

Richel J.C. Bilderbeek

find_first_regex_line *Find the first line that satisfies a regex*

Description

Find the first line that satisfies a regex

Usage

```
find_first_regex_line(lines, regex)
```

Arguments

lines	lines of tex
regex	the regex as text

Author(s)

Richel J.C. Bilderbeek

find_first_xml_opening_tag_line *Find the line number of the first section's opening tag*

Description

Find the line number of the first section's opening tag

Usage

```
find_first_xml_opening_tag_line(lines, section)
```

Arguments

lines	the lines of an XML text
section	the name of the XML section

Value

the line number's index (which is 1 for the first line) if the opening tag is found, else NA

Author(s)

Richel J.C. Bilderbeek

`find_last_xml_closing_tag_line` *Find the highest line number of a section's closing tag*

Description

Find the highest line number of a section's closing tag

Usage

```
find_last_xml_closing_tag_line(lines, section)
```

Arguments

`lines`	the lines of an XML text
`section`	the name of the XML section

Value

the line number's index (which is 1 for the first line) if the opening tag is found, else NA

`freq_equilibrium_to_xml` *Creates the `freq_equilibrium` as XML*

Description

Creates the `freq_equilibrium` as XML

Usage

```
freq_equilibrium_to_xml(freq_equilibrium, id)
```

Arguments

`freq_equilibrium`	a `freq_equilibrium` name
`id`	a site model's name

Value

the `freq_equilibrium` as XML

gamma_site_models_to_xml_prior_distr

Creates the gamma site models section in the distribution section of a BEAST2 XML parameter file

Description

Creates the gamma site models section in the distribution section of a BEAST2 XML parameter file

Usage

```
gamma_site_models_to_xml_prior_distr(site_models)
```

Arguments

site_models	one or more site models, as returned by create_site_model

Author(s)

Richel J.C. Bilderbeek

gamma_site_model_to_xml_prior_distr

Creates the gamma site models section in the distribution section of a BEAST2 XML parameter file

Description

Creates the gamma site models section in the distribution section of a BEAST2 XML parameter file

Usage

```
gamma_site_model_to_xml_prior_distr(site_model)
```

Arguments

site_model	a site model, as returned by create_site_model

Author(s)

Richel J.C. Bilderbeek

gamma_site_model_to_xml_state *Converts a gamma site model to XML, used in the `state` section*

Description

Converts a gamma site model to XML, used in the `state` section

Usage

```
gamma_site_model_to_xml_state(gamma_site_model, id)
```

Arguments

`gamma_site_model`	a gamma site model, as created by `create_gamma_site_model`)
`id`	the site model's ID

Value

the gamma_site model as XML text

Author(s)

Richel J.C. Bilderbeek

get_alignment_id *Get the alignment ID from a FASTA filename*

Description

Get the alignment ID from a FASTA filename

Usage

```
get_alignment_id(fasta_filename)
```

Arguments

`fasta_filename`	a FASTA filename. Use `get_fasta_filename` to obtain a testing FASTA filename.

Author(s)

Richel J.C. Bilderbeek

Examples

```
created <- get_alignment_id("/home/homer/anthus_aco_sub.fas")
expected <- "anthus_aco_sub"
testit::assert(created == expected)
```

get_alignment_ids *Get the alignment ID from one or more FASTA filenames*

Description

Get the alignment ID from one or more FASTA filenames

Usage

```
get_alignment_ids(fasta_filenames)
```

Arguments

fasta_filenames	One or more FASTA filenames. Use get_fasta_filename to obtain a testing FASTA filename.

Author(s)

Richel J.C. Bilderbeek

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

Description

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

Usage

```
get_beautier_path(filename)
```

Arguments

filename	the file's name, without the path

Value

the filename's full path

Author(s)

Richel J.C. Bilderbeek

See Also

for more files, use `get_beautier_paths`

Examples

```
testit::assert(is.character(get_beautier_path("test_output_0.fas")))
testit::assert(is.character(get_beautier_path("anthus_aco.fas")))
testit::assert(is.character(get_beautier_path("anthus_nd2.fas")))
```

get_beautier_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_beautier_paths(filenames)
```

Arguments

filenames	the files' names, without the path

Value

the filenames' full paths

Author(s)

Richel J.C. Bilderbeek

See Also

for one file, use `get_beautier_path`

Examples

```
testit::assert(
  length(
    get_beautier_paths(
      c("test_output_0.fas", "anthus_aco.fas", "anthus_nd2.fas")
    )
   ) == 3
 )
```

get_clock_models_ids *Collect the IDs of the list of clock models*

Description

Collect the IDs of the list of clock models

Usage

```
get_clock_models_ids(clock_models)
```

Arguments

clock_models	a list of one or more clock models, as returned by create_clock_model

Author(s)

Richel J.C. Bilderbeek

get_clock_model_name *Get the BEAUti name for a clock model*

Description

Get the BEAUti name for a clock model

Usage

```
get_clock_model_name(clock_model)
```

Arguments

clock_model	a clock model, as returned by create_clock_model

Author(s)

Richel J.C. Bilderbeek

Examples

```
strict <- create_strict_clock_model()
testit::assert(beautier:::get_clock_model_name(strict) == "StrictClock")
rln <- create_rln_clock_model()
testit::assert(beautier:::get_clock_model_name(rln) == "RelaxedClock")
```

`get_clock_model_names` *Get the clock model names*

Description

Get the clock model names

Usage

```
get_clock_model_names()
```

Value

the clock model names

Author(s)

Richel J.C. Bilderbeek

Examples

```
names <- beautier:::get_clock_model_names()
testit::assert("relaxed_log_normal" %in% names)
testit::assert("strict" %in% names)
```

`get_distr_names` *Get the distribution names*

Description

Get the distribution names

Usage

```
get_distr_names()
```

Value

the distribution names

Author(s)

Richel J.C. Bilderbeek

Examples

```
names <- beautier:::get_distr_names()
testit::assert("uniform" %in% names)
testit::assert("normal" %in% names)
testit::assert("one_div_x" %in% names)
testit::assert("log_normal" %in% names)
testit::assert("exponential" %in% names)
testit::assert("gamma" %in% names)
testit::assert("beta" %in% names)
testit::assert("laplace" %in% names)
testit::assert("inv_gamma" %in% names)
testit::assert("poisson" %in% names)
```

get_distr_n_params *Get the number of parameters a distribution uses*

Description

Get the number of parameters a distribution uses

Usage

```
get_distr_n_params(distr)
```

Arguments

distr	a distribution, as created by create_distr or (preferable) its named functions

Value

the number of parameters that distribution uses

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_distr_n_params(create_beta_distr()) == 2
)
testit::assert(
  beautier:::get_distr_n_params(create_exp_distr()) == 1
)
testit::assert(
  beautier:::get_distr_n_params(create_gamma_distr()) == 2
)
testit::assert(
  beautier:::get_distr_n_params(create_inv_gamma_distr()) == 2
)
testit::assert(
  beautier:::get_distr_n_params(create_laplace_distr()) == 2
)
testit::assert(
  beautier:::get_distr_n_params(create_log_normal_distr()) == 2
)
testit::assert(
  beautier:::get_distr_n_params(create_normal_distr()) == 2
)
testit::assert(
  beautier:::get_distr_n_params(create_one_div_x_distr()) == 0
)
testit::assert(
  beautier:::get_distr_n_params(create_poisson_distr()) == 1
)
testit::assert(
  beautier:::get_distr_n_params(create_uniform_distr()) == 0
)
```

get_fasta_filename *Get the path of a FASTA file used in testing*

Description

Get the path of a FASTA file used in testing

Usage

```
get_fasta_filename()
```

Value

the path of a FASTA file used in testing

Author(s)

Richel J.C. Bilderbeek

Examples

```
filename <- beautier::get_fasta_filename()
testit::assert(file.exists(filename))

create_beast2_input_file(
  input_filenames = filename,
  "my_beast.xml"
)
```

get_file_base_sans_ext

Get the filename's base without extension

Description

Get the filename's base without extension

Usage

```
get_file_base_sans_ext(filename)
```

Arguments

filename	A filename

Value

That filename without its full path and extension

Examples

```
testit::assert(
  beautier:::get_file_base_sans_ext("/home/richel/test.txt")
  == "test"
)
```

get_first_clock_model_index

Get the index of a clock model with a list of clock models

Description

Get the index of a clock model with a list of clock models

Usage

```
get_first_clock_model_index(clock_model, clock_models)
```

Arguments

clock_model	a clock model, as returned by create_clock_model
clock_models	a list of one or more clock models, as returned by create_clock_model

Value

the index of the first clock models with the same ID, NULL if such a clock model is absent

Author(s)

Richel J.C. Bilderbeek

Examples

```
a <- create_strict_clock_model(id = 1)
b <- create_rln_clock_model(id = 2)
ab <- list(a, b)
testit::assert(beautier:::get_first_clock_model_index(a, ab) == 1)
testit::assert(beautier:::get_first_clock_model_index(b, ab) == 2)
```

`get_freq_equilibrium_names` *Returns valid values for the* `freq_equilibrium` *argument*

Description

Returns valid values for the `freq_equilibrium` argument

Usage

```
get_freq_equilibrium_names()
```

Author(s)

Richel J.C. Bilderbeek

See Also

the `freq_equilibrium` argument is used in `create_gtr_site_model`, `create_hky_site_model`, and `create_tn93_site_model`

Examples

```
names <- beautier:::get_freq_equilibrium_names()
testit::assert("estimated" %in% names)
testit::assert("empirical" %in% names)
testit::assert("all_equal" %in% names)
```

get_has_non_strict_clock_model

Determines if there is at least one non-strict clock model in the list of one or more clock models

Description

Determines if there is at least one non-strict clock model in the list of one or more clock models

Usage

```
get_has_non_strict_clock_model(clock_models)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Author(s)

Richel J.C. Bilderbeek

get_id

Conclude the ID from a FASTA filename

Description

Conclude the ID from a FASTA filename

Usage

```
get_id(fasta_filename, capitalize_first_char_id = FALSE)
```

Arguments

`fasta_filename`	name of a FASTA file
`capitalize_first_char_id`	capitalize the first character of the ID

Value

the ID

Author(s)

Richel J.C. Bilderbeek

See Also

Use get_ids for one or more filenames

Examples

```
testit::assert(get_id("anthus_aco.fas") == "anthus_aco")
testit::assert(
  get_id("anthus_aco.fas", capitalize_first_char_id = TRUE)
  == "Anthus_aco")
```

get_ids *Conclude the IDs from one or more FASTA filenames*

Description

Conclude the IDs from one or more FASTA filenames

Usage

```
get_ids(fasta_filenames, capitalize_first_char_id = FALSE)
```

Arguments

fasta_filenames
: One or more FASTA filenames. Use get_fasta_filename to obtain a testing FASTA filename.

capitalize_first_char_id
: capitalize the first character of the IDs

Value

the IDs

Author(s)

Richel J.C. Bilderbeek

See Also

Use get_id for one filename

Examples

```
# Basic usage
testit::assert(get_ids(c("a.fas", "b.fas")) == c("a", "b"))

# Usage to create a BEAST2 XML file
fasta_filenames <- get_beautier_paths(
  c("anthus_aco.fas", "anthus_nd2.fas")
)
clock_models <- create_strict_clock_models(
  ids = get_ids(fasta_filenames)
)

create_beast2_input_file(
  fasta_filenames,
  "create_strict_clock_models.xml",
  clock_models = clock_models
)
testit::assert(file.exists("create_strict_clock_models.xml"))
```

get_n_taxa *Extract the number of taxa from a file*

Description

Extract the number of taxa from a file

Usage

```
get_n_taxa(filename)
```

Arguments

filename	name of a FASTA file

Value

the number of taxa

Author(s)

Richel J.C. Bilderbeek

get_operator_id_pre *Get the prefix of operator IDs*

Description

Get the prefix of operator IDs

Usage

```
get_operator_id_pre(tree_prior)
```

Arguments

tree_prior	a tree priors, as returned by create_tree_prior

Author(s)

Richel J.C. Bilderbeek

Examples

```
bd_pre <- beautier:::get_operator_id_pre(
  tree_prior = create_bd_tree_prior()
)
testthat::expect_equal(bd_pre, "BirthDeath")
```

get_param_names *Get the parameter names*

Description

Get the parameter names

Usage

```
get_param_names()
```

Value

the parameter names

Author(s)

Richel J.C. Bilderbeek

Examples

```
names <- beautier:::get_param_names()
testit::assert("alpha" %in% names)
testit::assert("beta" %in% names)
testit::assert("clock_rate" %in% names)
testit::assert("kappa_1" %in% names)
testit::assert("kappa_2" %in% names)
testit::assert("lambda" %in% names)
testit::assert("m" %in% names)
testit::assert("mean" %in% names)
testit::assert("mu" %in% names)
testit::assert("rate_ac" %in% names)
testit::assert("rate_ag" %in% names)
testit::assert("rate_at" %in% names)
testit::assert("rate_cg" %in% names)
testit::assert("rate_ct" %in% names)
testit::assert("rate_gt" %in% names)
testit::assert("s" %in% names)
testit::assert("scale" %in% names)
testit::assert("sigma" %in% names)
```

get_phylo_crown_age *Obtain the crown age of a phylony*

Description

Obtain the crown age of a phylony

Usage

```
get_phylo_crown_age(phylogeny)
```

Arguments

phylogeny	The phylogeny to obtain the crown age of

Value

the age of the phylogeny

Author(s)

Richel J.C. Bilderbeek

Examples

```
phylogeny <- ape::read.tree(text = "(a:15,b:15):1;")
created <- beautier:::get_phylo_crown_age(phylogeny = phylogeny)
testit::assert(created == 15)
```

get_site_models_n_distrs

Get the number of distributions a site model has

Description

Get the number of distributions a site model has

Usage

```
get_site_models_n_distrs(site_models)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`

Value

the number of distributions the site models have

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_site_models_n_distrs(list(create_gtr_site_model())) == 6
)
testit::assert(
  beautier:::get_site_models_n_distrs(list(create_hky_site_model())) == 2
)
testit::assert(
  beautier:::get_site_models_n_distrs(list(create_jc69_site_model())) == 1
)
testit::assert(
  beautier:::get_site_models_n_distrs(list(create_tn93_site_model())) == 3
)
```

get_site_models_n_params

Get the number of distributions one or more site models have

Description

Get the number of distributions one or more site models have

Usage

```
get_site_models_n_params(site_models)
```

Arguments

site_models	one or more site models, as returned by create_site_model

Value

the number of parameters the site models have

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_site_models_n_params(list(create_gtr_site_model())) == 11
)
testit::assert(
  beautier:::get_site_models_n_params(list(create_hky_site_model())) == 3
)
testit::assert(
  beautier:::get_site_models_n_params(list(create_jc69_site_model())) == 1
)
testit::assert(
  beautier:::get_site_models_n_params(list(create_tn93_site_model())) == 5
)
```

`get_site_model_names` *Get the site models' names*

Description

Get the site models' names

Usage

```
get_site_model_names()
```

Value

the site model names

Author(s)

Richel J.C. Bilderbeek

Examples

```
names <- beautier:::get_site_model_names()
testit::assert("JC69" %in% names)
testit::assert("HKY" %in% names)
testit::assert("TN93" %in% names)
testit::assert("GTR" %in% names)
```

`get_site_model_n_distrs` *Get the number of distributions a site model has*

Description

Get the number of distributions a site model has

Usage

```
get_site_model_n_distrs(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Value

the number of distributions a site model has

Author(s)

Richel J.C. Bilderbeek

Examples

```
# gamma site model, rates AC, AG, AT, CG and GT
testit::assert(
  beautier:::get_site_model_n_distrs(create_gtr_site_model()) == 6
)

# gamma site model, kappa
testit::assert(
  beautier:::get_site_model_n_distrs(create_hky_site_model()) == 2
)

# gamma site model
testit::assert(
  beautier:::get_site_model_n_distrs(create_jc69_site_model()) == 1
)

# gamma site model, kappa 1 and kappa 2
testit::assert(
  beautier:::get_site_model_n_distrs(create_tn93_site_model()) == 3
)
```

---

get_site_model_n_params

Get the number of distributions a site model has

Description

Get the number of distributions a site model has

Usage

```
get_site_model_n_params(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Value

the number of distributions a site model has

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_site_model_n_params(create_gtr_site_model()) == 11
)
testit::assert(
  beautier:::get_site_model_n_params(create_hky_site_model()) == 3
)
testit::assert(
  beautier:::get_site_model_n_params(create_jc69_site_model()) == 1
)
testit::assert(
  beautier:::get_site_model_n_params(create_tn93_site_model()) == 5
)
```

---

get_taxa_names *Extract the names of taxa from a file*

Description

Extract the names of taxa from a file

Usage

```
get_taxa_names(filename)
```

Arguments

filename	name of a FASTA file

Value

the taxa names

Author(s)

Richel J.C. Bilderbeek

Examples

```
created <- get_taxa_names(get_beautier_path("anthus_aco_sub.fas"))
expected <- c(
  "61430_aco", "626029_aco", "630116_aco", "630210_aco", "B25702_aco"
 )
testit::assert(created == expected)
```

get_tree_priors_n_distrs

Get the number of distributions a tree prior has

Description

Get the number of distributions a tree prior has

Usage

```
get_tree_priors_n_distrs(tree_priors)
```

Arguments

`tree_priors`	one or more tree priors, as returned by `create_tree_prior`

Value

the number of distributions a tree prior has

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_tree_priors_n_distrs(
    list(
      create_bd_tree_prior(), # has two distributions
      create_ccp_tree_prior() # has one distribution
    )
  ) == 3)
```

get_tree_priors_n_params

Get the number of parameters a list of tree priors has

Description

Get the number of parameters a list of tree priors has

Usage

```
get_tree_priors_n_params(tree_priors)
```

Arguments

`tree_priors`	one or more tree priors, as returned by `create_tree_prior`

Value

the number of parameters the tree priors have

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_tree_priors_n_params(
    list(
      create_bd_tree_prior(), # zero
      create_cep_tree_prior() # two
    )
  ) == 2
)
```

`get_tree_prior_names` *Get the tree prior names*

Description

Get the tree prior names

Usage

```
get_tree_prior_names()
```

Value

the tree prior names

Author(s)

Richel J.C. Bilderbeek

Examples

```
names <- beautier:::get_tree_prior_names()
testit::assert("birth_death" %in% names)
testit::assert("coalescent_bayesian_skyline" %in% names)
testit::assert("coalescent_constant_population" %in% names)
testit::assert("coalescent_exp_population" %in% names)
testit::assert("yule" %in% names)
```

get_tree_prior_n_distrs

Get the number of distributions a tree prior has

Description

Get the number of distributions a tree prior has

Usage

```
get_tree_prior_n_distrs(tree_prior)
```

Arguments

tree_prior	a tree priors, as returned by create_tree_prior

Value

the number of distributions a tree prior has

Author(s)

Richel J.C. Bilderbeek

Examples

```
# birth_rate_distr and death_rate_distr
testit::assert(
  beautier:::get_tree_prior_n_distrs(create_bd_tree_prior()) == 2
)

# none
testit::assert(
  beautier:::get_tree_prior_n_distrs(create_cbs_tree_prior()) == 0
)

# pop_size_distr
testit::assert(
  beautier:::get_tree_prior_n_distrs(create_ccp_tree_prior()) == 1
)

# pop_size_distr and growth_rate_distr
testit::assert(
  beautier:::get_tree_prior_n_distrs(create_cep_tree_prior()) == 2
)

# birth_rate_distr
testit::assert(
  beautier:::get_tree_prior_n_distrs(create_yule_tree_prior()) == 1
)
```

get_tree_prior_n_params

Get the number of parameters a tree prior has

Description

Get the number of parameters a tree prior has

Usage

```
get_tree_prior_n_params(tree_prior)
```

Arguments

`tree_prior`	a tree_prior, as created by `create_tree_prior`

Value

the number of parameters a tree prior has

Author(s)

Richel J.C. Bilderbeek

Examples

```
# birth_rate_distr is uniform, which has zero parameters
# death_rate_distr is uniform, which has zero parameters
testit::assert(
  beautier:::get_tree_prior_n_params(create_bd_tree_prior()) == 0
)

# no distributions, no parameters
testit::assert(
  beautier:::get_tree_prior_n_params(create_cbs_tree_prior()) == 0
)

# pop_size_distr is 1/x, which has zero parameters
testit::assert(
  beautier:::get_tree_prior_n_params(create_ccp_tree_prior()) == 0
)

# pop_size_distr is 1/x, which has zero parameters
# growth_rate_distr is Laplace, which has two parameters
testit::assert(
  beautier:::get_tree_prior_n_params(create_cep_tree_prior()) == 2
)

# birth_rate_distr is uniform, which has zero parameters
```

```
testit::assert(
  beautier:::get_tree_prior_n_params(create_yule_tree_prior()) == 0
)
```

get_unlinked_clock_models

Get the first clock model of each ID

Description

Get the first clock model of each ID

Usage

```
get_unlinked_clock_models(clock_models)
```

Arguments

`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Author(s)

Richel J.C. Bilderbeek

get_unlinked_site_models

Get the first site model of each ID

Description

Get the first site model of each ID

Usage

```
get_unlinked_site_models(site_models)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`

Author(s)

Richel J.C. Bilderbeek

get_unlinked_tree_priors *Get the first tree prior of each ID*

Description

Get the first tree prior of each ID

Usage

```
get_unlinked_tree_priors(tree_priors)
```

Arguments

`tree_priors`	one or more tree priors, as returned by `create_tree_prior`

Author(s)

Richel J.C. Bilderbeek

get_xml_closing_tag *Get the XML closing tag*

Description

Get the XML closing tag

Usage

```
get_xml_closing_tag(text)
```

Arguments

`text`	lines of XML to extract the XML closing tag from

Value

the closing tag if found, else NA

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_xml_closing_tag("<my_tag text=something></my_tag>")
  == "my_tag"
)
testit::assert(
  is.na(
    beautier:::get_xml_closing_tag("<my_tag text=something/>")
  )
)
testit::assert(is.na(beautier:::get_xml_closing_tag("no_xml")))
```

`get_xml_opening_tag` *Get the XML opening tag*

Description

Get the XML opening tag

Usage

```
get_xml_opening_tag(text)
```

Arguments

`text`	text to be determined to be valid

Value

the openin tag if found, else NA

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::get_xml_opening_tag("<my_tag text=something/>")
  == "my_tag"
)
testit::assert(is.na(beautier:::get_xml_opening_tag("no_xml")))
```

has_shared_rln_clock_models *Are there any shared RLN clock models?*

Description

Are there any shared RLN clock models?

Usage

```
has_shared_rln_clock_models(clock_models)
```

Arguments

clock_models	a list of one or more clock models, as returned by create_clock_model

Note

this must be FALSE, as BEAUti rejects this as well

Author(s)

Richel J.C. Bilderbeek

has_xml_closing_tag *Is an XML opening tag with value 'section' present amongst the lines of the text?*

Description

Is an XML opening tag with value 'section' present amongst the lines of the text?

Usage

```
has_xml_closing_tag(lines, section)
```

Arguments

lines	lines of the XML text
section	the XML section

has_xml_opening_tag *Is an XML opening tag with value 'section' present amongst the lines of the text?*

Description

Is an XML opening tag with value 'section' present amongst the lines of the text?

Usage

```
has_xml_opening_tag(lines, section = NA)
```

Arguments

lines	lines of an XML text
section	if NA, this function returns TRUE if there is any XML opening tag. If section is set to a certain word, this function returns TRUE if that tag matches section

has_xml_short_closing_tag *Is an XML closing tag with short closing text at the end of the text?*

Description

Is an XML closing tag with short closing text at the end of the text?

Usage

```
has_xml_short_closing_tag(lines)
```

Arguments

lines	lines of an XML text

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(beautier:::has_xml_short_closing_tag("<my_tag id=1/>"))
testit::assert(
  !beautier:::has_xml_short_closing_tag(
    "<my_tag id=1>text</my_tag>"
  )
)
```

`indent` *Indent text for a certain number of spaces If the text is only whitespace, leave it as such*

Description

Indent text for a certain number of spaces If the text is only whitespace, leave it as such

Usage

```
indent(text, n_spaces)
```

Arguments

`text`	the text to indent
`n_spaces`	the number of spaces to add before the text

Value

the indented text

Author(s)

Richel J.C. Bilderbeek

`init_bd_tree_prior` *Initializes a Birth-Death tree prior*

Description

Initializes a Birth-Death tree prior

Usage

```
init_bd_tree_prior(bd_tree_prior, distr_id, param_id)
```

Arguments

`bd_tree_prior`	a Birth-Death tree prior, as created by `create_bd_tree_prior`
`distr_id`	a distributions' ID
`param_id`	a parameter's ID

Value

an initialized Birth-Death tree prior

Author(s)

Richel J.C. Bilderbeek

`init_ccp_tree_prior` *Initializes a Coalescent Constant Population tree prior*

Description

Initializes a Coalescent Constant Population tree prior

Usage

```
init_ccp_tree_prior(ccp_tree_prior, distr_id, param_id)
```

Arguments

`ccp_tree_prior`	a Coalescent Constant Population tree prior, as returned by `create_ccp_tree_prior`
`distr_id`	a distributions' ID
`param_id`	a parameter's ID

Value

an initialized Coalescent Constant Population tree prior

Author(s)

Richel J.C. Bilderbeek

`init_cep_tree_prior` *Initializes a Coalescent Exponential Population tree prior*

Description

Initializes a Coalescent Exponential Population tree prior

Usage

```
init_cep_tree_prior(cep_tree_prior, distr_id, param_id)
```

Arguments

`cep_tree_prior`	a cep tree prior, as returned by `create_cep_tree_prior`
`distr_id`	a distributions' ID
`param_id`	a parameter's ID

Value

an initialized Coalescent Exponential Population tree prior

Author(s)

Richel J.C. Bilderbeek

`init_clock_models` *Initializes all clock models*

Description

Initializes all clock models

Usage

```
init_clock_models(fasta_filenames, clock_models, distr_id = 0, param_id = 0)
```

Arguments

`fasta_filenames`	One or more FASTA filenames. Use `get_fasta_filename` to obtain a testing FASTA filename.
`clock_models`	a list of one or more clock models, as returned by `create_clock_model`
`distr_id`	the first distributions' ID
`param_id`	the first parameter's ID

Value

a list of initialized clock models

Author(s)

Richel J.C. Bilderbeek

init_distr *Initializes a distribution*

Description

Initializes a distribution

Usage

```
init_distr(distr, distr_id = 0, param_id = 0)
```

Arguments

`distr`	a distribution, using `create_distr`
`distr_id`	the first distribution's ID
`param_id`	the first parameter's ID

Value

an initialized distribution

Author(s)

Richel J.C. Bilderbeek

init_gamma_site_model *Initializes all site models*

Description

Initializes all site models

Usage

```
init_gamma_site_model(gamma_site_model, distr_id = 0, param_id = 0)
```

Arguments

`gamma_site_model`	a site model's gamma site model, as returned by `create_gamma_site_model`
`distr_id`	the first distributions' ID
`param_id`	the first parameter's ID

Value

a list of initialized site models

Author(s)

Richel J.C. Bilderbeek

init_gtr_site_model *Initializes a GTR site model*

Description

Initializes a GTR site model

Usage

```
init_gtr_site_model(gtr_site_model, distr_id = 0, param_id = 0)
```

Arguments

gtr_site_model	a GTR site model, as returned by create_gtr_site_model
distr_id	a distributions' ID
param_id	a parameter's ID

Value

an initialized GTR site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
gtr_site_model <- create_gtr_site_model()
testit::assert(!beautier:::is_init_gtr_site_model(gtr_site_model))
gtr_site_model <- beautier:::init_gtr_site_model(gtr_site_model)
testit::assert(beautier:::is_init_gtr_site_model(gtr_site_model))
```

init_hky_site_model *Initializes an HKY site model*

Description

Initializes an HKY site model

Usage

```
init_hky_site_model(hky_site_model, distr_id = 0, param_id = 0)
```

Arguments

hky_site_model	an HKY site model, as returned by create_hky_site_model
distr_id	a distributions' ID
param_id	a parameter's ID

Value

an initialized HKY site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
hky_site_model <- create_hky_site_model()
testit::assert(!beautier:::is_init_hky_site_model(hky_site_model))
hky_site_model <- beautier:::init_hky_site_model(hky_site_model)
testit::assert(beautier:::is_init_hky_site_model(hky_site_model))
```

init_jc69_site_model *Initializes a JC69 site model*

Description

Initializes a JC69 site model

Usage

```
init_jc69_site_model(jc69_site_model, distr_id = 0, param_id = 0)
```

Arguments

`jc69_site_model`	a JC69 site model, as returned by `create_jc69_site_model`
`distr_id`	a distributions' ID
`param_id`	a parameter's ID

Value

an initialized HKY site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
hky_site_model <- create_hky_site_model()
testit::assert(!beautier:::is_init_hky_site_model(hky_site_model))
hky_site_model <- beautier:::init_hky_site_model(hky_site_model)
testit::assert(beautier:::is_init_hky_site_model(hky_site_model))
```

`init_mrca_priors` *Initializes all mrca priors*

Description

Initializes all mrca priors

Usage

```
init_mrca_priors(mrca_priors, distr_id = 0, param_id = 0)
```

Arguments

`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`
`distr_id`	the first distributions' ID
`param_id`	the first parameter's ID

Value

a list of initialized mrca priors

Author(s)

Richel J.C. Bilderbeek

init_param *Initializes a parameter*

Description

Initializes a parameter

Usage

```
init_param(param, id)
```

Arguments

param	a parameter, using create_param
id	the parameter's ID. Will be ignored if the parameter already has an ID

Value

an initialized parameter

Author(s)

Richel J.C. Bilderbeek

init_rln_clock_model *Initializes a Relaxed Log-Normal clock model*

Description

Initializes a Relaxed Log-Normal clock model

Usage

```
init_rln_clock_model(rln_clock_model, distr_id, param_id)
```

Arguments

rln_clock_model	a Relaxed Log-Normal clock model, as returned by create_rln_clock_model
distr_id	a distributions' ID
param_id	a parameter's ID

Value

an initialized Relaxed Log-Normal clock model

Author(s)

Richel J.C. Bilderbeek

init_site_models *Initializes all site models*

Description

Initializes all site models

Usage

```
init_site_models(site_models, ids, distr_id = 0, param_id = 0)
```

Arguments

site_models	one or more site models, as returned by create_site_model
ids	one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with get_ids)
distr_id	the first distributions' ID
param_id	the first parameter's ID

Value

a list of initialized site models

Author(s)

Richel J.C. Bilderbeek

init_strict_clock_model *Initializes a strict clock model*

Description

Initializes a strict clock model

Usage

```
init_strict_clock_model(strict_clock_model, distr_id, param_id)
```

Arguments

`strict_clock_model`
: a strict clock model, as returned by `create_strict_clock_model`

`distr_id`
: a distributions' ID

`param_id`
: a parameter's ID

Value

an initialized strict clock model

Author(s)

Richel J.C. Bilderbeek

Examples

```
strict_clock_model <- create_strict_clock_model()
```

`init_tn93_site_model` *Initializes a TN93 site model*

Description

Initializes a TN93 site model

Usage

```
init_tn93_site_model(tn93_site_model, distr_id = 0, param_id = 0)
```

Arguments

`tn93_site_model`
: a TN93 site model, as returned by `create_tn93_site_model`

`distr_id`
: a distributions' ID

`param_id`
: a parameter's ID

Value

an initialized TN93 site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
tn93_site_model <- create_tn93_site_model()
testit::assert(!beautier:::is_init_tn93_site_model(tn93_site_model))
tn93_site_model <- beautier:::init_tn93_site_model(tn93_site_model)
testit::assert(beautier:::is_init_tn93_site_model(tn93_site_model))
```

`init_tree_priors` *Initializes all tree priors*

Description

Initializes all tree priors

Usage

```
init_tree_priors(tree_priors, ids, distr_id = 0, param_id = 0)
```

Arguments

`tree_priors`	one or more tree priors, as returned by `create_tree_prior`
`ids`	one or more alignments' IDs. IDs can be extracted from their FASTA filesnames with `get_ids`)
`distr_id`	the first distributions' ID
`param_id`	the first parameter's ID

Value

a list of initialized tree priors

Author(s)

Richel J.C. Bilderbeek

`init_yule_tree_prior` *Initializes a Yule tree prior*

Description

Initializes a Yule tree prior

Usage

```
init_yule_tree_prior(yule_tree_prior, distr_id, param_id)
```

Arguments

`yule_tree_prior`	a Yule tree_prior, as created by `create_yule_tree_prior`
`distr_id`	a distributions' ID
`param_id`	a parameter's ID

Value

an initialized Yule tree prior

Author(s)

Richel J.C. Bilderbeek

`interspace` *Puts spaces in between the lines*

Description

Puts spaces in between the lines

Usage

```
interspace(lines)
```

Arguments

`lines`	lines of text

`is_alpha_param` *Determine if the object is a valid alpha parameter*

Description

Determine if the object is a valid alpha parameter

Usage

```
is_alpha_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid alpha parameter

Value

TRUE if x is a valid alpha parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_bd_tree_prior` *Determine if the object is a valid Birth Death tree prior as returned by* `create_bd_tree_prior`

Description

Determine if the object is a valid Birth Death tree prior as returned by `create_bd_tree_prior`

Usage

```
is_bd_tree_prior(x)
```

Arguments

`x`	an object, to be determined if it is a valid birth death tree prior

Value

TRUE if x is a valid birth death tree prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_beta_distr` *Determine if the object is a valid beta distribution, as created by* `create_beta_distr`

Description

Determine if the object is a valid beta distribution, as created by `create_beta_distr`

Usage

```
is_beta_distr(x)
```

Arguments

`x`	an object, to be determined if it is a valid beta distribution,

Value

TRUE if x is a valid beta distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use `is_distr` to see if x is any distribution

`is_beta_param` *Determine if the object is a valid beta parameter*

Description

Determine if the object is a valid beta parameter

Usage

```
is_beta_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid beta parameter

Value

TRUE if x is a valid beta parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_cbs_tree_prior` *Determine if the object is a valid constant coalescent Bayesian skyline prior as returned by `create_cbs_tree_prior`*

Description

Determine if the object is a valid constant coalescent Bayesian skyline prior as returned by `create_cbs_tree_prior`

Usage

```
is_cbs_tree_prior(x)
```

Arguments

`x`	an object, to be determined if it is a valid constant coalescent Bayesian skyline prior

Value

TRUE if x is a valid constant coalescent Bayesian skyline prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_ccp_tree_prior` *Determine if the object is a valid constant coalescence population tree prior, as returned by `create_ccp_tree_prior`*

Description

Determine if the object is a valid constant coalescence population tree prior, as returned by `create_ccp_tree_prior`

Usage

```
is_ccp_tree_prior(x)
```

Arguments

`x`	an object, to be determined if it is a valid constant coalescence population tree prior

Value

TRUE if x is a valid constant coalescence population tree prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_cep_tree_prior` *Determine if the object is a valid coalescent exponential population tree prior as returned by* `create_cep_tree_prior`

Description

Determine if the object is a valid coalescent exponential population tree prior as returned by `create_cep_tree_prior`

Usage

```
is_cep_tree_prior(x)
```

Arguments

`x`	an object, to be determined if it is a valid constant coalescent exponential population tree prior

Value

TRUE if x is a valid coalescent exponentialpopulation tree prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_clock_model` *Determine if the object is a valid clock_model*

Description

Determine if the object is a valid clock_model

Usage

```
is_clock_model(x)
```

Arguments

`x`	an object, to be determined if it is a clock_model

Value

TRUE if the clock_model is a valid clock_model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

see `create_clock_model` for an overview of functions to create valid clock model

`is_clock_model_name` *Determines if the name is a valid clock model name*

Description

Determines if the name is a valid clock model name

Usage

```
is_clock_model_name(name)
```

Arguments

`name`	the name to be tested

Value

TRUE if the name is a valid clock_model name, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(beautier:::is_clock_model_name("relaxed_log_normal"))
testit::assert(beautier:::is_clock_model_name("strict"))
```

is_clock_rate_param *Determine if the object is a valid clock_rate parameter*

Description

Determine if the object is a valid clock_rate parameter

Usage

```
is_clock_rate_param(x)
```

Arguments

x	an object, to be determined if it is a valid clock_rate parameter

Value

TRUE if x is a valid clock_rate parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_distr *Determine if the object is a valid distribution*

Description

Determine if the object is a valid distribution

Usage

```
is_distr(x)
```

Arguments

x	an object, to be determined if it is a valid distribution

Value

TRUE if x is a valid distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use is_beta_distr, is_exp_distr, is_gamma_distr, is_inv_gamma_distr, is_laplace_distr, is_log_normal_distr, is_normal_distr, is_one_div_x_distr, is_poisson_distr, or is_uniform_distr, to check for more specific distribution

---

is_distr_name *Determines if the name is a valid distribution name*

Description

Determines if the name is a valid distribution name

Usage

```
is_distr_name(name)
```

Arguments

name	the name to be tested

Value

TRUE if the name is a valid distribution name, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(beautier:::is_distr_name("uniform"))
testit::assert(beautier:::is_distr_name("normal"))
testit::assert(beautier:::is_distr_name("one_div_x"))
testit::assert(beautier:::is_distr_name("log_normal"))
testit::assert(beautier:::is_distr_name("exponential"))
testit::assert(beautier:::is_distr_name("gamma"))
testit::assert(beautier:::is_distr_name("beta"))
testit::assert(beautier:::is_distr_name("laplace"))
testit::assert(beautier:::is_distr_name("inv_gamma"))
testit::assert(beautier:::is_distr_name("poisson"))
```

is_exp_distr *Determine if the object is a valid exponential distribution as created by* `create_exp_distr`

Description

Determine if the object is a valid exponential distribution as created by `create_exp_distr`

Usage

```
is_exp_distr(x)
```

Arguments

`x`	an object, to be determined if it is a valid exponential distribution

Value

TRUE if x is a valid exponential distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use `is_distr` to see if x is any distribution

is_freq_equilibrium_name *Checks if* `name` *is a valid* `freq_equilibrium` *argument value*

Description

Checks if `name` is a valid `freq_equilibrium` argument value

Usage

```
is_freq_equilibrium_name(name)
```

Arguments

`name`	the name to check if it is a valid `freq_equilibrium` argument value

Author(s)

Richel J.C. Bilderbeek

See Also

the freq_equilibrium argument is used by create_gtr_site_model, create_hky_site_model, and create_tn93_site_model

Examples

```
testit::assert(beautier:::is_freq_equilibrium_name("estimated"))
testit::assert(beautier:::is_freq_equilibrium_name("empirical"))
testit::assert(beautier:::is_freq_equilibrium_name("all_equal"))
```

is_gamma_distr *Determine if the object is a valid gamma distribution, as created by create_gamma_distr*

Description

Determine if the object is a valid gamma distribution, as created by create_gamma_distr

Usage

```
is_gamma_distr(x)
```

Arguments

x	an object, to be determined if it is a valid gamma distribution

Value

TRUE if x is a valid gamma distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use is_distr to see if x is any distribution

is_gamma_site_model *Is object x a gamma site model?*

Description

Is object x a gamma site model?

Usage

```
is_gamma_site_model(x)
```

Arguments

x	the object to be determined if it is a valid gamma site object

Value

TRUE if x is a valid gamma site object, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
gamma_site_model <- create_gamma_site_model()
testit::assert(beautier:::is_gamma_site_model(gamma_site_model))
```

is_gtr_site_model *Determine if the object is a valid GTR site model, as created by create_gtr_site_model*

Description

Determine if the object is a valid GTR site model, as created by `create_gtr_site_model`

Usage

```
is_gtr_site_model(x)
```

Arguments

x	an object, to be determined if it is a valid GTR site model

Value

TRUE if x is a valid GTR site model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
gtr_site_model <- create_gtr_site_model()
testit::assert(beautier:::is_gtr_site_model(gtr_site_model))
```

is_hky_site_model *Determine if the object is a valid HKY site model, as created by* create_hky_site_model

Description

Determine if the object is a valid HKY site model, as created by create_hky_site_model

Usage

```
is_hky_site_model(x)
```

Arguments

x	an object, to be determined if it is a valid HKY site model

Value

TRUE if x is a valid HKY site model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
hky_site_model <- create_hky_site_model()
testit::assert(beautier:::is_hky_site_model(hky_site_model))
```

is_id *Determine if the object is a valid ID*

Description

Determine if the object is a valid ID

Usage

```
is_id(x)
```

Arguments

x	an object, to be determined if it is a valid ID

Value

TRUE if x is a valid ID, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

to check multiple IDs, use are_ids

is_index_of_first_shared_clock_model *Is the index the first of a series of clock models that are shared?*

Description

Is the index the first of a series of clock models that are shared?

Usage

```
is_index_of_first_shared_clock_model(i, clock_models)
```

Arguments

i	the index of the clock model within clock_models
clock_models	a list of one or more clock models, as returned by create_clock_model

Author(s)

Richel J.C. Bilderbeek

`is_index_of_non_first_shared_clock_model` *Is the index the first of a series of clock models that are shared?*

Description

Is the index the first of a series of clock models that are shared?

Usage

```
is_index_of_non_first_shared_clock_model(i, clock_models)
```

Arguments

`i`	the index of the clock model within `clock_models`
`clock_models`	a list of one or more clock models, as returned by `create_clock_model`

Author(s)

Richel J.C. Bilderbeek

`is_init_bd_tree_prior` *Determine if x is an initialized Birth-Death tree_prior object*

Description

Determine if x is an initialized Birth-Death tree_prior object

Usage

```
is_init_bd_tree_prior(x)
```

Arguments

`x`	the object to check if it is an initialized Birth-Death tree prior object

Value

TRUE if x is an initialized Birth-Death tree_prior object

Author(s)

Richel J.C. Bilderbeek

is_init_beta_distr *Determine if x is an initialized beta distribution object as created by* create_beta_distr

Description

Determine if x is an initialized beta distribution object as created by create_beta_distr

Usage

```
is_init_beta_distr(x)
```

Arguments

x	the object to check if it is an initialized beta distribution object

Value

TRUE if x is an initialized beta distribution object

Author(s)

Richel J.C. Bilderbeek

is_init_cbs_tree_prior *Determine if x is an initialized Coalescent Bayesian Skyline tree_prior object*

Description

Determine if x is an initialized Coalescent Bayesian Skyline tree_prior object

Usage

```
is_init_cbs_tree_prior(x)
```

Arguments

x	the object to check if it is an initialized Coalescent Bayesian Skyline tree prior object

Value

TRUE if x is an initialized Coalescent Bayesian Skyline tree prior object

Author(s)

Richel J.C. Bilderbeek

---

is_init_ccp_tree_prior *Determine if x is an initialized Coalescent Constant Population tree_prior object*

Description

Determine if x is an initialized Coalescent Constant Population tree_prior object

Usage

```
is_init_ccp_tree_prior(x)
```

Arguments

x	the object to check if it is an initialized Coalescent Constant Population tree prior object

Value

TRUE if x is an initialized Coalescent Constant Population tree prior object

Author(s)

Richel J.C. Bilderbeek

---

is_init_cep_tree_prior *Determine if x is an initialized Coalescent Exponential Population tree_prior object*

Description

Determine if x is an initialized Coalescent Exponential Population tree_prior object

Usage

```
is_init_cep_tree_prior(x)
```

Arguments

x	the object to check if it is an initialized Coalescent Exponential Population tree prior object

Value

TRUE if x is an initialized Coalescent Exponential Population tree prior object

Author(s)

Richel J.C. Bilderbeek

`is_init_clock_model` *Determine if x is an initialized clock_model object, as created by* `create_clock_model`

Description

Determine if x is an initialized clock_model object, as created by `create_clock_model`

Usage

```
is_init_clock_model(x)
```

Arguments

`x`	the object to check if it is an initialized clock_models object

Value

TRUE if x is an initialized clock_model object

Author(s)

Richel J.C. Bilderbeek

`is_init_distr` *Determine if x is an initialized distribution object as created by* `create_distr`

Description

Determine if x is an initialized distribution object as created by `create_distr`

Usage

```
is_init_distr(x)
```

Arguments

`x`	the object to check if it is an initialized distribution object

Value

TRUE if x is an initialized distribution object

Author(s)

Richel J.C. Bilderbeek

`is_init_exp_distr` *Determine if x is an initialized exponential distribution object as created by* `create_exp_distr`

Description

Determine if x is an initialized exponential distribution object as created by `create_exp_distr`

Usage

```
is_init_exp_distr(x)
```

Arguments

`x`	the object to check if it is an initialized exponential distribution object

Value

TRUE if x is an initialized exponential distribution object

Author(s)

Richel J.C. Bilderbeek

`is_init_gamma_distr` *Determine if x is an initialized gamma distribution object*

Description

Determine if x is an initialized gamma distribution object

Usage

```
is_init_gamma_distr(x)
```

Arguments

`x`	the object to check if it is an initialized gamma distribution object

Value

TRUE if x is an initialized gamma distribution object

Author(s)

Richel J.C. Bilderbeek

`is_init_gamma_site_model`

Determine if x is an initialized gamma site model, as created by `create_gamma_site_model`

Description

Determine if x is an initialized gamma site model, as created by `create_gamma_site_model`

Usage

```
is_init_gamma_site_model(x)
```

Arguments

x	the object to check if it is an initialized gamma site_models object

Value

TRUE if x is an initialized gamma site model

Author(s)

Richel J.C. Bilderbeek

`is_init_gtr_site_model`

Determine if x is an initialized GTR site model as created by `create_gtr_site_model`

Description

Determine if x is an initialized GTR site model as created by `create_gtr_site_model`

Usage

```
is_init_gtr_site_model(x)
```

Arguments

`x`	the object to check if it is an initialized GTR site model

Value

TRUE if x is an initialized GTR site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
gtr_site_model <- create_gtr_site_model()
testit::assert(!beautier:::is_init_gtr_site_model(gtr_site_model))
gtr_site_model <- beautier:::init_gtr_site_model(gtr_site_model)
testit::assert(beautier:::is_init_gtr_site_model(gtr_site_model))
```

`is_init_hky_site_model` *Determine if x is an initialized hky site model as created by* `create_hky_site_model`

Description

Determine if x is an initialized hky site model as created by `create_hky_site_model`

Usage

```
is_init_hky_site_model(x)
```

Arguments

`x`	the object to check if it is an initialized HKY site model

Value

TRUE if x is an initialized HKY site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
hky_site_model <- create_hky_site_model()
testit::assert(!beautier:::is_init_hky_site_model(hky_site_model))
hky_site_model <- beautier:::init_hky_site_model(hky_site_model)
testit::assert(beautier:::is_init_hky_site_model(hky_site_model))
```

is_init_inv_gamma_distr

Determine if x is an initialized inv_gamma distribution object as created by `create_inv_gamma_distr`

Description

Determine if x is an initialized inv_gamma distribution object as created by `create_inv_gamma_distr`

Usage

```
is_init_inv_gamma_distr(x)
```

Arguments

`x`	the object to check if it is an initialized inv_gamma distribution object

Value

TRUE if x is an initialized inv_gamma distribution object

Author(s)

Richel J.C. Bilderbeek

is_init_jc69_site_model

Determine if x is an initialized JC69 site model as created by `create_jc69_site_model`

Description

Determine if x is an initialized JC69 site model as created by `create_jc69_site_model`

Usage

```
is_init_jc69_site_model(x)
```

Arguments

`x`	the object to check if it is an initialized JC69 site model

Value

TRUE if x is an initialized JC69 site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
jc69_site_model <- create_jc69_site_model()
testit::assert(!beautier:::is_init_jc69_site_model(jc69_site_model))
jc69_site_model <- beautier:::init_jc69_site_model(jc69_site_model)
testit::assert(beautier:::is_init_jc69_site_model(jc69_site_model))
```

is_init_laplace_distr *Determine if x is an initialized laplace distribution object as created by* create_laplace_distr

Description

Determine if x is an initialized laplace distribution object as created by create_laplace_distr

Usage

```
is_init_laplace_distr(x)
```

Arguments

x	the object to check if it is an initialized laplace distribution object

Value

TRUE if x is an initialized laplace distribution object

Author(s)

Richel J.C. Bilderbeek

is_init_log_normal_distr *Determine if x is an initialized log_normal distribution object as created by* create_log_normal_distr

Description

Determine if x is an initialized log_normal distribution object as created by create_log_normal_distr

Usage

```
is_init_log_normal_distr(x)
```

Arguments

`x`	the object to check if it is an initialized log_normal distribution object

Value

TRUE if x is an initialized log_normal distribution object

Author(s)

Richel J.C. Bilderbeek

`is_init_mrca_prior` *Determine if x is an initialized mrca_prior objects*

Description

Determine if x is an initialized mrca_prior objects

Usage

```
is_init_mrca_prior(x)
```

Arguments

`x`	the object to check if it is an initialized mrca_priors object

Value

TRUE if x is an initialized mrca_prior object

Author(s)

Richel J.C. Bilderbeek

`is_init_normal_distr` *Determine if x is an initialized normal distribution object as created by `create_normal_distr`*

Description

Determine if x is an initialized normal distribution object as created by `create_normal_distr`

Usage

```
is_init_normal_distr(x)
```

Arguments

`x`	the object to check if it is an initialized normal distribution object

Value

TRUE if x is an initialized normal distribution object

Author(s)

Richel J.C. Bilderbeek

`is_init_one_div_x_distr` *Determine if x is an initialized one_div_x distribution object as created by `create_one_div_x_distr`*

Description

Determine if x is an initialized one_div_x distribution object as created by `create_one_div_x_distr`

Usage

```
is_init_one_div_x_distr(x)
```

Arguments

`x`	the object to check if it is an initialized one_div_x distribution object

Value

TRUE if x is an initialized one_div_x distribution object

Author(s)

Richel J.C. Bilderbeek

is_init_param *Determine if x is an initialized parameter, as created by* create_param

Description

Determine if x is an initialized parameter, as created by create_param

Usage

```
is_init_param(x)
```

Arguments

x	the object to check if it is an initialized parameter

Value

TRUE if x is an initialized parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_init_poisson_distr *Determine if x is an initialized poisson distribution object as created by* create_poisson_distr

Description

Determine if x is an initialized poisson distribution object as created by create_poisson_distr

Usage

```
is_init_poisson_distr(x)
```

Arguments

x	the object to check if it is an initialized poisson distribution object

Value

TRUE if x is an initialized poisson distribution object

Author(s)

Richel J.C. Bilderbeek

is_init_rln_clock_model *Determine if x is an initialized relaxed log-normal clock_model object*

Description

Determine if x is an initialized relaxed log-normal clock_model object

Usage

```
is_init_rln_clock_model(rln_clock_model)
```

Arguments

rln_clock_model	a Relaxed Log-Normal clock model, as returned by create_rln_clock_model

Value

TRUE if x is an initialized relaxed log-normal clock_model object, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_init_site_model *Determine if x is an initialized site model, as created by create_site_model*

Description

Determine if x is an initialized site model, as created by create_site_model

Usage

```
is_init_site_model(x)
```

Arguments

x	the object to check if it is an initialized site_models object

Value

TRUE if x is an initialized site model

Author(s)

Richel J.C. Bilderbeek

`is_init_strict_clock_model` *Determine if x is an initialized strict clock_model object*

Description

Determine if x is an initialized strict clock_model object

Usage

```
is_init_strict_clock_model(strict_clock_model)
```

Arguments

`strict_clock_model`	a strict clock model, as returned by `create_strict_clock_model`

Value

TRUE if x is an initialized strict clock_model object

Author(s)

Richel J.C. Bilderbeek

`is_init_tn93_site_model` *Determine if x is an initialized tn93 site model as created by* `create_tn93_site_model`

Description

Determine if x is an initialized tn93 site model as created by `create_tn93_site_model`

Usage

```
is_init_tn93_site_model(x)
```

Arguments

`x`	the object to check if it is an initialized TN93 site model

Value

TRUE if x is an initialized TN93 site model

Author(s)

Richel J.C. Bilderbeek

Examples

```
tn93_site_model <- create_tn93_site_model()
testit::assert(!beautier:::is_init_tn93_site_model(tn93_site_model))
tn93_site_model <- beautier:::init_tn93_site_model(tn93_site_model)
testit::assert(beautier:::is_init_tn93_site_model(tn93_site_model))
```

`is_init_tree_prior` *Determine if x is an initialized tree_prior objects*

Description

Determine if x is an initialized tree_prior objects

Usage

```
is_init_tree_prior(x)
```

Arguments

`x`	the object to check if it is an initialized tree_priors object

Value

TRUE if x is an initialized tree_prior object

Author(s)

Richel J.C. Bilderbeek

`is_init_uniform_distr` *Determine if x is an initialized uniform distribution object as created by `create_uniform_distr`*

Description

Determine if x is an initialized uniform distribution object as created by `create_uniform_distr`

Usage

```
is_init_uniform_distr(x)
```

Arguments

x	the object to check if it is an initialized uniform distribution object

Value

TRUE if x is an initialized uniform distribution object

Author(s)

Richel J.C. Bilderbeek

is_init_yule_tree_prior *Determine if x is an initialized Yule tree_prior object*

Description

Determine if x is an initialized Yule tree_prior object

Usage

```
is_init_yule_tree_prior(x)
```

Arguments

x	the object to check if it is an initialized Yule tree prior object

Value

TRUE if x is an initialized Yule tree_prior object

Author(s)

Richel J.C. Bilderbeek

is_inv_gamma_distr *Determine if the object is a valid inv_gamma distribution, as created by create_inv_gamma_distr*

Description

Determine if the object is a valid inv_gamma distribution, as created by create_inv_gamma_distr

Usage

```
is_inv_gamma_distr(x)
```

Arguments

x	an object, to be determined if it is a valid inv_gamma distribution

Value

TRUE if x is a valid inv_gamma distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use is_distr to see if x is any distribution

is_in_patterns *Create a random beta distribution*

Description

Create a random beta distribution

Usage

```
is_in_patterns(line, patterns)
```

Arguments

line	a line of text
patterns	one or more regular expression patterns

Value

TRUE if there is at least one match found

Author(s)

Richel J.C. Bilderbeek

is_jc69_site_model *Determine if the object is a valid JC69 site model*

Description

Determine if the object is a valid JC69 site model

Usage

```
is_jc69_site_model(x)
```

Arguments

x	an object, to be determined if it is a valid JC69 site model

Value

TRUE if x is a valid JC69 site model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
jc69_site_model <- create_jc69_site_model()
testit::assert(beautier:::is_jc69_site_model(jc69_site_model))
```

is_kappa_1_param *Determine if the object is a valid kappa 1 parameter*

Description

Determine if the object is a valid kappa 1 parameter

Usage

```
is_kappa_1_param(x)
```

Arguments

x	an object, to be determined if it is a valid kappa 1 parameter

Value

TRUE if x is a valid kappa 1 parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

kappa 1 parameters are returned by `create_kappa_1_param`

Examples

```
kappa_1_param <- create_kappa_1_param()
testit::assert(beautier:::is_kappa_1_param(kappa_1_param))
```

`is_kappa_2_param` *Determine if the object is a valid kappa 2 parameter*

Description

Determine if the object is a valid kappa 2 parameter

Usage

```
is_kappa_2_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid kappa 2 parameter

Value

TRUE if x is a valid kappa_2 parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

kappa 2 parameters are returned by `create_kappa_2_param`

Examples

```
kappa_2_param <- create_kappa_2_param()
testit::assert(beautier:::is_kappa_2_param(kappa_2_param))
```

is_lambda_param *Determine if the object is a valid lambda parameter*

Description

Determine if the object is a valid lambda parameter

Usage

```
is_lambda_param(x)
```

Arguments

x	an object, to be determined if it is a valid lambda parameter

Value

TRUE if x is a valid lambda parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

lambda parameters are returned by create_lambda_param

Examples

```
lambda_param <- create_lambda_param()
testit::assert(beautier:::is_lambda_param(lambda_param))
```

is_laplace_distr *Determine if the object is a valid laplace distribution, as created by create_laplace_distr*

Description

Determine if the object is a valid laplace distribution, as created by create_laplace_distr

Usage

```
is_laplace_distr(x)
```

Arguments

x	an object, to be determined if it is a valid laplace distribution

Value

TRUE if x is a valid laplace distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use `is_distr` to see if x is any distribution

Examples

```
laplace_distr <- create_laplace_distr()
testit::assert(beautier:::is_laplace_distr(laplace_distr))
```

`is_log_normal_distr` *Determine if the object is a valid log-normal distribution, as created by* `create_log_normal_distr`

Description

Determine if the object is a valid log-normal distribution, as created by `create_log_normal_distr`

Usage

```
is_log_normal_distr(x)
```

Arguments

`x`	an object, to be determined if it is a valid log-normal distribution

Value

TRUE if x is a valid log-normal distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use `is_distr` to see if x is any distribution

Examples

```
log_normal_distr <- create_log_normal_distr()

input_fasta_filename <- beautier::get_beautier_path("anthus_aco.fas")
create_beast2_input_file(
  input_filenames = input_fasta_filename,
  "my_beast.xml",
  tree_priors = create_yule_tree_prior(
    birth_rate_distr = log_normal_distr
  )
)
testit::assert(file.exists("my_beast.xml"))
```

is_mcmc *Determine if the object is a valid mcmc object*

Description

Determine if the object is a valid mcmc object

Usage

```
is_mcmc(x)
```

Arguments

x	an object, to be determined if it is a valid mcmc object

Value

TRUE if x is a valid mcmc object, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

Use `create_mcmc`

is_mean_param *Determine if the object is a valid mean parameter*

Description

Determine if the object is a valid mean parameter

Usage

```
is_mean_param(x)
```

Arguments

x	an object, to be determined if it is a valid mean parameter, as created by `create_mean_param`)

Value

TRUE if x is a valid mean parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_misc_options *Determine if the object is a valid misc_options*

Description

Determine if the object is a valid misc_options

Usage

```
is_misc_options(x)
```

Arguments

x	an object, to be determined if it is a misc_options

Value

TRUE if the object is a valid misc_options, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
misc_options <- create_misc_options()
testit::assert(beautier:::is_misc_options(misc_options))
```

is_mrca_prior *Determine of the object is an MRCA prior, as returned by create_mrca_prior*

Description

Determine of the object is an MRCA prior, as returned by `create_mrca_prior`

Usage

```
is_mrca_prior(x)
```

Arguments

x	object to be determined if it is an MRCA prior

Value

TRUE if x is an MRCA prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_mu_param *Determine if the object is a valid mu parameter*

Description

Determine if the object is a valid mu parameter

Usage

```
is_mu_param(x)
```

Arguments

x	an object, to be determined if it is a valid mu parameter

Value

TRUE if x is a valid mu parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

`create_mu_param` creates a mu parameter

Examples

```
mu_param <- create_mu_param()
testit::assert(beautier:::is_mu_param(mu_param))
```

`is_m_param` *Determine if the object is a valid m parameter*

Description

Determine if the object is a valid m parameter

Usage

```
is_m_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid m parameter

Value

TRUE if x is a valid m parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_normal_distr *Determine if the object is a valid normal distribution as created by create_normal_distr*

Description

Determine if the object is a valid normal distribution as created by `create_normal_distr`

Usage

```
is_normal_distr(x)
```

Arguments

x	an object, to be determined if it is a valid normal distribution

Value

TRUE if x is a valid normal distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use `is_distr` to see if x is any distribution

is_one_div_x_distr *Determine if the object is a valid 1/x distribution, as created by create_one_div_x_distr*

Description

Determine if the object is a valid 1/x distribution, as created by `create_one_div_x_distr`

Usage

```
is_one_div_x_distr(x)
```

Arguments

x	an object, to be determined if it is a valid 1/x distribution

Value

TRUE if x is a valid 1/x distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use `is_distr` to see if x is any distribution

`is_one_na` *Determines if x is one NA*

Description

Determines if x is one NA

Usage

```
is_one_na(x)
```

Arguments

`x`	the object to be determined if it is one NA

Value

TRUE if x is one NA, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(beautier:::is_one_na(NA))
testit::assert(!beautier:::is_one_na(NULL))
testit::assert(!beautier:::is_one_na(42))
testit::assert(!beautier:::is_one_na("Hello"))
testit::assert(!beautier:::is_one_na(3.14))
testit::assert(!beautier:::is_one_na(c(NA, NA)))
```

`is_param` *Determine if the object is a valid parameter*

Description

Determine if the object is a valid parameter

Usage

```
is_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid parameter, as created by `create_param`)

Value

TRUE if x is a valid parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

`is_param_name` *Determines if the name is a valid parameter name*

Description

Determines if the name is a valid parameter name

Usage

```
is_param_name(name)
```

Arguments

`name`	the name to be tested

Value

TRUE if the name is a valid parameter name, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(beautier:::is_param_name("alpha"))
testit::assert(beautier:::is_param_name("beta"))
testit::assert(beautier:::is_param_name("clock_rate"))
testit::assert(beautier:::is_param_name("kappa_1"))
testit::assert(beautier:::is_param_name("kappa_2"))
testit::assert(beautier:::is_param_name("lambda"))
testit::assert(beautier:::is_param_name("m"))
testit::assert(beautier:::is_param_name("mean"))
testit::assert(beautier:::is_param_name("mu"))
testit::assert(beautier:::is_param_name("rate_ac"))
testit::assert(beautier:::is_param_name("rate_ag"))
testit::assert(beautier:::is_param_name("rate_at"))
testit::assert(beautier:::is_param_name("rate_cg"))
testit::assert(beautier:::is_param_name("rate_ct"))
testit::assert(beautier:::is_param_name("rate_gt"))
testit::assert(beautier:::is_param_name("s"))
testit::assert(beautier:::is_param_name("scale"))
testit::assert(beautier:::is_param_name("sigma"))
```

`is_phylo` *Checks if the input is a phylogeny*

Description

Checks if the input is a phylogeny

Usage

```
is_phylo(x)
```

Arguments

`x`	input to be checked

Value

TRUE or FALSE

Author(s)

Richel J.C. Bilderbeek

is_poisson_distr *Determine if the object is a valid poisson distribution as created by create_poisson_distr*

Description

Determine if the object is a valid poisson distribution as created by create_poisson_distr

Usage

```
is_poisson_distr(x)
```

Arguments

x	an object, to be determined if it is a valid poisson distribution

Value

TRUE if x is a valid poisson distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use is_distr to see if x is any distribution

is_rate_ac_param *Determine if the object is a valid 'rate AC' parameter*

Description

Determine if the object is a valid 'rate AC' parameter

Usage

```
is_rate_ac_param(x)
```

Arguments

x	an object, to be determined if it is a valid 'rate AC' parameter

Value

TRUE if x is a valid 'rate AC' parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

`create_rate_ac_param` creates a ’rate AC’ parameter

Examples

```
rate_ac_param <- create_rate_ac_param()
testit::assert(beautier:::is_rate_ac_param(rate_ac_param))
```

`is_rate_ag_param` *Determine if the object is a valid ’rate AG’ parameter*

Description

Determine if the object is a valid ’rate AG’ parameter

Usage

```
is_rate_ag_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid ’rate AG’ parameter

Value

TRUE if x is a valid ’rate AG’ parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

`create_rate_ag_param` creates a ’rate AG’ parameter

Examples

```
rate_ag_param <- create_rate_ag_param()
```

is_rate_at_param *Determine if the object is a valid 'rate AT' parameter*

Description

Determine if the object is a valid 'rate AT' parameter

Usage

```
is_rate_at_param(x)
```

Arguments

x	an object, to be determined if it is a valid 'rate AT' parameter

Value

TRUE if x is a valid 'rate AT' parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

`create_rate_at_param` creates a 'rate AT' parameter

Examples

```
rate_at_param <- create_rate_at_param()
testit::assert(beautier:::is_rate_at_param(rate_at_param))
```

is_rate_cg_param *Determine if the object is a valid 'rate CG' parameter*

Description

Determine if the object is a valid 'rate CG' parameter

Usage

```
is_rate_cg_param(x)
```

Arguments

x	an object, to be determined if it is a valid 'rate CG' parameter

Value

TRUE if x is a valid 'rate CG' parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

`create_rate_cg_param` creates a 'rate CG' parameter

Examples

```
rate_cg_param <- create_rate_cg_param()
testit::assert(beautier:::is_rate_cg_param(rate_cg_param))
```

`is_rate_ct_param` *Determine if the object is a valid 'rate CT' parameter*

Description

Determine if the object is a valid 'rate CT' parameter

Usage

```
is_rate_ct_param(x)
```

Arguments

`x`	an object, to be determined if it is a valid 'rate CT' parameter

Value

TRUE if x is a valid 'rate CG' parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

`create_rate_ct_param` creates a 'rate CT' parameter

Examples

```
rate_ct_param <- create_rate_ct_param()
testit::assert(beautier:::is_rate_ct_param(rate_ct_param))
```

is_rate_gt_param *Determine if the object is a valid 'rate GT' parameter*

Description

Determine if the object is a valid 'rate GT' parameter

Usage

```
is_rate_gt_param(x)
```

Arguments

x	an object, to be determined if it is a valid 'rate GT' parameter

Value

TRUE if x is a valid 'rate GT' parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

create_rate_gt_param creates a 'rate GT' parameter

Examples

```
rate_gt_param <- create_rate_gt_param()
testit::assert(beautier:::is_rate_gt_param(rate_gt_param))
```

is_rln_clock_model *Determine if the object is a valid relaxed log normal clock model*

Description

Determine if the object is a valid relaxed log normal clock model

Usage

```
is_rln_clock_model(x)
```

Arguments

x	an object, to be determined if it is a valid relaxed log normal clock model, as created by create_rln_clock_model)

Value

TRUE if x is a valid relaxed log normal clock model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

create_clock_model shows an overview of functions to create a clock model

Examples

```
rln_clock_model <- create_rln_clock_model()
testit::assert(beautier:::is_rln_clock_model(rln_clock_model))

strict_clock_model <- create_strict_clock_model()
testit::assert(beautier:::is_strict_clock_model(strict_clock_model))
```

is_scale_param *Determine if the object is a valid scale parameter*

Description

Determine if the object is a valid scale parameter

Usage

```
is_scale_param(x)
```

Arguments

x	an object, to be determined if it is a valid scale parameter

Value

TRUE if x is a valid scale parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_sigma_param *Determine if the object is a valid sigma parameter*

Description

Determine if the object is a valid sigma parameter

Usage

```
is_sigma_param(x)
```

Arguments

x	an object, to be determined if it is a valid sigma parameter

Value

TRUE if x is a valid sigma parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

is_site_model *Determine if the object is a valid site_model*

Description

Determine if the object is a valid site_model

Usage

```
is_site_model(x)
```

Arguments

x	an object, to be determined if it is a site_model

Value

TRUE if the site_model is a valid site_model, FALSE otherwise

See Also

A site model can be created using `create_site_model`

Examples

```
# site models
testit::assert(beautier:::is_site_model(create_gtr_site_model()))
testit::assert(beautier:::is_site_model(create_hky_site_model()))
testit::assert(beautier:::is_site_model(create_jc69_site_model()))
testit::assert(beautier:::is_site_model(create_tn93_site_model()))

# other models
testit::assert(!beautier:::is_site_model(create_strict_clock_model()))
testit::assert(!beautier:::is_site_model(create_bd_tree_prior()))
testit::assert(!beautier:::is_site_model(create_mcmc()))
```

is_site_model_name *Determines if the name is a valid site_model name*

Description

Determines if the name is a valid site_model name

Usage

```
is_site_model_name(name)
```

Arguments

name	the name to be tested

Value

TRUE if the name is a valid site_model name, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(beautier:::is_site_model_name("JC69"))
testit::assert(beautier:::is_site_model_name("HKY"))
testit::assert(beautier:::is_site_model_name("TN93"))
testit::assert(beautier:::is_site_model_name("GTR"))
```

is_strict_clock_model *Determine if the object is a valid strict clock model, as returned by create_strict_clock_model*

Description

Determine if the object is a valid strict clock model, as returned by create_strict_clock_model

Usage

```
is_strict_clock_model(x)
```

Arguments

x	an object, to be determined if it is a valid strict clock model

Value

TRUE if x is a valid strict clock model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

create_clock_model shows an overview of functions to create a clock model

Examples

```
strict_clock_model <- create_strict_clock_model()

# rln: Relaxed Log-Normal
rln_clock_model <- create_rln_clock_model()
testit::assert(!beautier:::is_strict_clock_model(rln_clock_model))
```

is_s_param *Determine if the object is a valid s parameter*

Description

Determine if the object is a valid s parameter

Usage

```
is_s_param(x)
```

Arguments

x an object, to be determined if it is a valid s parameter

Value

TRUE if x is a valid s parameter, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

---

is_tn93_site_model *Determine if the object is a valid TN93 site model,*

Description

Determine if the object is a valid TN93 site model,

Usage

```
is_tn93_site_model(x)
```

Arguments

x an object, to be determined if it is a valid TN93 site model, as created by create_tn93_site_model

Value

TRUE if x is a valid TN93 site model, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
create_beast2_input_file(
  input_filenames = get_fasta_filename(),
  "beast.xml",
  site_models = create_tn93_site_model()
)
```

is_tree_prior *Determine if an object is a valid tree prior*

Description

Determine if an object is a valid tree prior

Usage

```
is_tree_prior(x)
```

Arguments

x	an object

Value

TRUE if x is a valid tree_prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

tree priors can be created by `create_tree_prior`)

is_tree_prior_name *Determines if the name is a valid tree prior name*

Description

Determines if the name is a valid tree prior name

Usage

```
is_tree_prior_name(name)
```

Arguments

name	the name to be tested

Value

TRUE if the name is a valid tree_prior name, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

Examples

```
testit::assert(
  beautier:::is_tree_prior_name("birth_death")
)
testit::assert(
  beautier:::is_tree_prior_name("coalescent_bayesian_skyline")
)
testit::assert(
  beautier:::is_tree_prior_name("coalescent_constant_population")
)
testit::assert(
  beautier:::is_tree_prior_name("coalescent_exp_population")
)
testit::assert(
  beautier:::is_tree_prior_name("yule")
)
```

is_uniform_distr *Determine if the object is a valid uniform distribution as created by* create_uniform_distr

Description

Determine if the object is a valid uniform distribution as created by create_uniform_distr

Usage

```
is_uniform_distr(x)
```

Arguments

x	an object, to be determined if it is a valid uniform distribution

Value

TRUE if x is a valid uniform distribution, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

See Also

use is_distr to see if x is any distribution

is_xml — *Checks if the text is a valid XML node, that is, it has a opening and matching closing tag*

Description

Checks if the text is a valid XML node, that is, it has a opening and matching closing tag

Usage

```
is_xml(text)
```

Arguments

`text`	text to be determined to be valid

Author(s)

Richel J.C. Bilderbeek

is_yule_tree_prior — *Determine if the object is a valid Yule tree prior, as returned by `create_yule_tree_prior`*

Description

Determine if the object is a valid Yule tree prior, as returned by `create_yule_tree_prior`

Usage

```
is_yule_tree_prior(x)
```

Arguments

`x`	an object, to be determined if it is a valid Yule tree prior

Value

TRUE if x is a valid Yule tree prior, FALSE otherwise

Author(s)

Richel J.C. Bilderbeek

mcmc_to_xml_run *Converts an MCMC object to the run section's XML*

Description

Converts an MCMC object to the run section's XML

Usage

```
mcmc_to_xml_run(mcmc)
```

Arguments

mcmc	one mcmc object, as returned by `create_mcmc`

Value

the XML as text

Author(s)

Richel J.C. Bilderbeek

Examples

```
xml <- beautier:::mcmc_to_xml_run(create_mcmc())
testit::assert(xml ==
  "<run id=\"mcmc\" spec=\"MCMC\" chainLength=\"10000000\">"
)
```

mrca_priors_to_xml_operators *Creates the XML of a list of one or more MRCA priors, as used in the `operators` section*

Description

Creates the XML of a list of one or more MRCA priors, as used in the `operators` section

Usage

```
mrca_priors_to_xml_operators(mrca_priors, has_non_strict_clock_model = FALSE)
```

Arguments

mrca_priors
: a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

has_non_strict_clock_model
: boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

Value

the MRCA priors as XML text

Author(s)

Richel J.C. Bilderbeek

---

mrca_priors_to_xml_prior_distr

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Description

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Usage

```
mrca_priors_to_xml_prior_distr(mrca_priors, has_non_strict_clock_model)
```

Arguments

mrca_priors
: a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`

has_non_strict_clock_model
: boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

---

mrca_priors_to_xml_state

Converts one or more MRCA priors to the `state` *section of the XML as text*

Description

Converts one or more MRCA priors to the `state` section of the XML as text

Usage

```
mrca_priors_to_xml_state(mrca_priors, has_non_strict_clock_model = FALSE)
```

Arguments

`mrca_priors`	a list of one or more Most Recent Common Ancestor priors, as returned by `create_mrca_prior`
`has_non_strict_clock_model`	boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

Value

lines of XML text, without indentation nor `state` tags

Author(s)

Richel J.C. Bilderbeek

mrca_priors_to_xml_tracelog

Creates the mrca priors' XML for the tracelog section

Description

Creates the mrca priors' XML for the tracelog section

Usage

```
mrca_priors_to_xml_tracelog(mrca_priors, has_non_strict_clock_model = FALSE)
```

Arguments

mrca_priors	a list of one or more Most Recent Common Ancestor priors, as returned by create_mrca_prior
has_non_strict_clock_model	boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

Author(s)

Richel J.C. Bilderbeek

See Also

the complete tracelog section is created by create_beast2_input_tracelog

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

mrca_prior_to_xml_lh_distr

Converts an MRCA prior to the branchRateModel *section of the XML as text*

Description

Converts an MRCA prior to the branchRateModel section of the XML as text

Usage

```
mrca_prior_to_xml_lh_distr(mrca_prior, has_non_strict_clock_model = FALSE)
```

Arguments

`mrca_prior` a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`

`has_non_strict_clock_model`
boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#     </distribution>
#     <distribution id="likelihood" ...>
#       HERE, where the ID of the distribution is 'likelihood'
#     </distribution>
# </distribution>
```

`mrca_prior_to_xml_operators`
Creates the XML of an MRCA prior, as used in the `operators` *section*

Description

Creates the XML of an MRCA prior, as used in the `operators` section

Usage

```
mrca_prior_to_xml_operators(mrca_prior, fixed_crown_age = FALSE,
  has_non_strict_clock_model = FALSE, is_first = TRUE)
```

Arguments

`mrca_prior` a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`

`fixed_crown_age`
determines if the phylogeny's crown age is fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, the crown age is fixed to the crown age of the initial phylogeny.

`has_non_strict_clock_model`
boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

`is_first` is this the first MRCA prior?

Value

the mrca prior as XML text

Author(s)

Richel J.C. Bilderbeek

---

mrca_prior_to_xml_prior_distr

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Description

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Usage

```
mrca_prior_to_xml_prior_distr(mrca_prior, has_non_strict_clock_model = FALSE,
  taxa_names_with_ids = NULL, is_first = TRUE)
```

Arguments

mrca_prior	a Most Recent Common Ancestor prior, as returned by create_mrca_prior
has_non_strict_clock_model	boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model
taxa_names_with_ids	taxa names that already have received an ID. Causes the XML to idref these
is_first	is this the first MRCA prior?

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

mrca_prior_to_xml_state *Creates the XML of an MRCA prior, as used in the `state` section*

Description

Creates the XML of an MRCA prior, as used in the `state` section

Usage

```
mrca_prior_to_xml_state(mrca_prior, has_non_strict_clock_model = FALSE,
  is_first = TRUE)
```

Arguments

`mrca_prior`	a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`
`has_non_strict_clock_model`	boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model
`is_first`	is this the first MRCA prior?

Value

the tree prior as XML text

Author(s)

Richel J.C. Bilderbeek

mrca_prior_to_xml_taxonset *Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='*

Description

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Usage

```
mrca_prior_to_xml_taxonset(mrca_prior, taxa_names_with_ids = NULL)
```

Arguments

`mrca_prior` a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`

`taxa_names_with_ids` taxa names that already have received an ID. Causes the XML to `idref` these

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <taxonset id="all" spec="TaxonSet">
#     <taxon id="626029_aco" spec="Taxon"/>
#     <taxon id="630116_aco" spec="Taxon"/>
#     <taxon id="630210_aco" spec="Taxon"/>
#     <taxon id="B25702_aco" spec="Taxon"/>
#     <taxon id="61430_aco" spec="Taxon"/>
# </taxonset>
```

mrca_prior_to_xml_tracelog

Creates the mrca prior's XML for the tracelog section

Description

Creates the mrca prior's XML for the tracelog section

Usage

```
mrca_prior_to_xml_tracelog(mrca_prior, has_non_strict_clock_model = FALSE,
  is_first = TRUE)
```

Arguments

`mrca_prior` a Most Recent Common Ancestor prior, as returned by `create_mrca_prior`

`has_non_strict_clock_model` boolean to indicate that the is already at least one non-strict (i.e. relaxed log-normal) clock model

`is_first` is this the first MRCA prior?

Author(s)

Richel J.C. Bilderbeek

See Also

all mrca priors' tracelog section is created by `mrca_priors_to_xml_tracelog`

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

`parameter_to_xml` *Converts a parameter to XML*

Description

Converts a parameter to XML

Usage

```
parameter_to_xml(parameter)
```

Arguments

`parameter`	a distibution, as created by `create_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

Examples

```
  xml <- beautier:::parameter_to_xml(create_alpha_param(id = 1))
  testit::assert(length(xml) == 1)
  testit::assert(nchar(xml) > 1)
```

`parameter_to_xml_alpha`
Converts an alpha parameter to XML

Description

Converts an alpha parameter to XML

Usage

```
parameter_to_xml_alpha(parameter)
```

Arguments

`parameter`	an alpha parameter, as created by `create_alpha_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_beta` *Converts a beta parameter to XML*

Description

Converts a beta parameter to XML

Usage

```
parameter_to_xml_beta(parameter)
```

Arguments

`parameter`	a beta parameter, as created by `create_beta_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

parameter_to_xml_clock_rate

Converts a clockRate parameter to XML

Description

Converts a clockRate parameter to XML

Usage

```
parameter_to_xml_clock_rate(parameter)
```

Arguments

parameter	a clockRate parameter, as created by create_clock_rate_param)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

parameter_to_xml_kappa_1

Converts a kappa 1 parameter to XML

Description

Converts a kappa 1 parameter to XML

Usage

```
parameter_to_xml_kappa_1(parameter)
```

Arguments

parameter	a kappa 1 parameter, as created by create_kappa_1_param)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

parameter_to_xml_kappa_2

Converts a kappa 2 parameter to XML

Description

Converts a kappa 2 parameter to XML

Usage

```
parameter_to_xml_kappa_2(parameter)
```

Arguments

`parameter`	a kappa 2 parameter, as created by `create_kappa_2_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

parameter_to_xml_lambda

Converts a lambda parameter to XML

Description

Converts a lambda parameter to XML

Usage

```
parameter_to_xml_lambda(parameter)
```

Arguments

`parameter`	a lambda parameter, as created by `create_lambda_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

parameter_to_xml_m *Converts a m parameter to XML*

Description

Converts a m parameter to XML

Usage

```
parameter_to_xml_m(parameter)
```

Arguments

parameter	a m parameter, as created by create_m_param)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

parameter_to_xml_mean *Converts a mean parameter to XML*

Description

Converts a mean parameter to XML

Usage

```
parameter_to_xml_mean(parameter)
```

Arguments

parameter	a mean parameter, as created by create_mean_param)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_mu` *Converts a mu parameter to XML*

Description

Converts a mu parameter to XML

Usage

```
parameter_to_xml_mu(parameter)
```

Arguments

`parameter`	a mu parameter, as created by `create_mu_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_rate_ac` *Converts a 'rate AC' parameter to XML*

Description

Converts a 'rate AC' parameter to XML

Usage

```
parameter_to_xml_rate_ac(parameter, which_name = "state_node")
```

Arguments

`parameter`	a 'rate AC' parameter, as created by `create_rate_ac_param`)
`which_name`	the name, can be `state_node` or `rate_name`

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_rate_ag`
Converts a 'rate AG' parameter to XML

Description

Converts a 'rate AG' parameter to XML

Usage

```
parameter_to_xml_rate_ag(parameter, which_name = "state_node")
```

Arguments

`parameter`	a 'rate AG' parameter, as created by `create_rate_ag_param`)
`which_name`	the name, can be `state_node` or `rate_name`

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_rate_at`
Converts a 'rate AT' parameter to XML

Description

Converts a 'rate AT' parameter to XML

Usage

```
parameter_to_xml_rate_at(parameter, which_name = "state_node")
```

Arguments

`parameter`	a 'rate AT' parameter, as created by `create_rate_at_param`)
`which_name`	the name, can be `state_node` or `rate_name`

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

---

parameter_to_xml_rate_cg

Converts a 'rate CG' parameter to XML

Description

Converts a 'rate CG' parameter to XML

Usage

```
parameter_to_xml_rate_cg(parameter, which_name = "state_node")
```

Arguments

parameter	a 'rate CG' parameter, as created by create_rate_cg_param)
which_name	the name, can be state_node or rate_name

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

---

parameter_to_xml_rate_ct

Converts a 'rate CT' parameter to XML

Description

Converts a 'rate CT' parameter to XML

Usage

```
parameter_to_xml_rate_ct(parameter, which_name = "state_node")
```

Arguments

parameter	a 'rate CT' parameter, as created by create_rate_ct_param)
which_name	the name, can be state_node or rate_name

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_rate_gt` *Converts a 'rate GT' parameter to XML*

Description

Converts a 'rate GT' parameter to XML

Usage

```
parameter_to_xml_rate_gt(parameter, which_name = "state_node")
```

Arguments

`parameter`	a 'rate GT' parameter, as created by `create_rate_gt_param`)
`which_name`	the name, can be `state_node` or `rate_name`

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`parameter_to_xml_s` *Converts a s parameter to XML*

Description

Converts a s parameter to XML

Usage

```
parameter_to_xml_s(parameter)
```

Arguments

`parameter`	a s parameter, as created by `create_s_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

---

parameter_to_xml_scale

Converts a scale parameter to XML

Description

Converts a scale parameter to XML

Usage

```
parameter_to_xml_scale(parameter)
```

Arguments

`parameter`	a scale parameter, as created by `create_scale_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

---

parameter_to_xml_sigma

Converts a sigma parameter to XML

Description

Converts a sigma parameter to XML

Usage

```
parameter_to_xml_sigma(parameter)
```

Arguments

`parameter`	a sigma parameter, as created by `create_sigma_param`)

Value

the parameter as XML text

Author(s)

Richel J.C. Bilderbeek

`phylo_to_xml_state` *Creates the XML of a phylogeny, as used in the `state` section*

Description

Creates the XML of a phylogeny, as used in the `state` section

Usage

```
phylo_to_xml_state(phylo, id)
```

Arguments

`phylo`	the phylogeny. If NA, a random phylogeny is used
`id`	the ID of the alignment

Value

the random phylogeny as XML text

Author(s)

Richel J.C. Bilderbeek

`remove_multiline` *Remove consequetive lines*

Description

Remove consequetive lines

Usage

```
remove_multiline(text, lines_to_remove)
```

Arguments

`text`	lines of characters
`lines_to_remove`	lines of character that need to be removed from text

Author(s)

Richel J.C. Bilderbeek

rln_clock_model_to_xml_mean_rate_prior *Used by clock_models_to_xml_prior_distr*

Description

Used by `clock_models_to_xml_prior_distr`

Usage

```
rln_clock_model_to_xml_mean_rate_prior(rln_clock_model)
```

Arguments

`rln_clock_model`	a Relaxed Log-Normal clock model, as returned by `create_rln_clock_model`

Author(s)

Richel J.C. Bilderbeek

rnd_phylo_to_xml_init *Creates the XML of a random phylogeny, as used in the `init` section*

Description

Creates the XML of a random phylogeny, as used in the `init` section

Usage

```
rnd_phylo_to_xml_init(id)
```

Arguments

`id`	an alignment's IDs. An ID can be extracted from its FASTA filename with `get_ids`)

Value

the phylogeny as XML text

Author(s)

Richel J.C. Bilderbeek

rnd_phylo_to_xml_state *Creates the XML of a random phylogeny, as used in the state section*

Description

Creates the XML of a random phylogeny, as used in the `state` section

Usage

```
rnd_phylo_to_xml_state(id)
```

Arguments

`id`	an alignment's IDs. An ID can be extracted from its FASTA filename with `get_ids`)

Value

the random phylogeny as XML text

Author(s)

Richel J.C. Bilderbeek

site_models_to_xml_prior_distr *Represent the site models as XML*

Description

Represent the site models as XML

Usage

```
site_models_to_xml_prior_distr(site_models)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

---

site_models_to_xml_state

Converts one or more clock models to the `state` *section of the XML as text*

Description

Converts one or more clock models to the `state` section of the XML as text

Usage

```
site_models_to_xml_state(site_models)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`

Value

lines of XML text, without indentation nor `state` tags

Author(s)

Richel J.C. Bilderbeek

---

site_models_to_xml_tracelog

Creates the site models' XML for the tracelog section

Description

Creates the site models' XML for the tracelog section

Usage

```
site_models_to_xml_tracelog(site_models)
```

Arguments

site_models one or more site models, as returned by create_site_model

Author(s)

Richel J.C. Bilderbeek

See Also

the complete tracelog section is created by create_beast2_input_tracelog

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

---

site_model_to_xml_lh_distr

Converts a site model to XML, used in the `siteModel` *section*

Description

Converts a site model to XML, used in the `siteModel` section

Usage

```
site_model_to_xml_lh_distr(site_model)
```

Arguments

site_model a site model, as returned by create_site_model

Value

the site model as XML text

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#     </distribution>
#     <distribution id="likelihood" ...>
#       HERE, where the ID of the distribution is 'likelihood'
#     </distribution>
# </distribution>
```

site_model_to_xml_operators

Converts a site model to XML, used in the `operators` *section*

Description

Converts a site model to XML, used in the `operators` section

Usage

```
site_model_to_xml_operators(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Value

the site model as XML text

Author(s)

Richel J.C. Bilderbeek

site_model_to_xml_prior_distr

Converts a site model to XML, used in the `prior` *section*

Description

Converts a site model to XML, used in the `prior` section

Usage

```
site_model_to_xml_prior_distr(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Value

the site model as XML text

Author(s)

Richel J.C. Bilderbeek

site_model_to_xml_state

Converts a site model to XML, used in the `state` section

Description

Converts a site model to XML, used in the `state` section

Usage

```
site_model_to_xml_state(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Value

the site model as XML text

Author(s)

Richel J.C. Bilderbeek

site_model_to_xml_subst_model

Converts a site model to XML, used in the `substModel` section

Description

Converts a site model to XML, used in the `substModel` section

Usage

```
site_model_to_xml_subst_model(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Value

the site model as XML text

Author(s)

Richel J.C. Bilderbeek

site_model_to_xml_tracelog

Creates the site model's XML for the tracelog section

Description

Creates the site model's XML for the tracelog section

Usage

```
site_model_to_xml_tracelog(site_model)
```

Arguments

`site_model`	a site model, as returned by `create_site_model`

Author(s)

Richel J.C. Bilderbeek

See Also

all site models' tracelog section is created by `site_models_to_xml_tracelog`

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

tree_models_to_xml_tracelog

Creates the tree models' XML for the tracelog section

Description

Creates the tree models' XML for the tracelog section

Usage

```
tree_models_to_xml_tracelog(site_models)
```

Arguments

`site_models`	one or more site models, as returned by `create_site_model`

Note

use site_models just because it contains all IDs

Author(s)

Richel J.C. Bilderbeek

See Also

the complete tracelog section is created by `create_beast2_input_tracelog`

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

`tree_priors_to_xml_operators` *Creates the XML of a list of one or more tree priors, as used in the* `operators` *section*

Description

Creates the XML of a list of one or more tree priors, as used in the `operators` section

Usage

```
tree_priors_to_xml_operators(tree_priors, fixed_crown_ages = rep(FALSE, times
  = length(tree_priors)))
```

Arguments

`tree_priors` one or more tree priors, as returned by `create_tree_prior`

`fixed_crown_ages` one or more booleans to determine if the phylogenies' crown ages are fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, the crown age is fixed to the crown age of the initial phylogeny.

Value

the tree priors as XML text

Author(s)

Richel J.C. Bilderbeek

tree_priors_to_xml_prior_distr

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Description

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Usage

```
tree_priors_to_xml_prior_distr(tree_priors)
```

Arguments

tree_priors	one or more tree priors, as returned by create_tree_prior

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

tree_priors_to_xml_state

Converts one or more tree priors to the `state` section of the XML as text

Description

Converts one or more tree priors to the `state` section of the XML as text

Usage

```
tree_priors_to_xml_state(tree_priors)
```

Arguments

`tree_priors`	one or more tree priors, as returned by `create_tree_prior`

Value

lines of XML text, without indentation nor `state` tags

Author(s)

Richel J.C. Bilderbeek

`tree_priors_to_xml_tracelog` *Creates the tree priors' XML for the tracelog section*

Description

Creates the tree priors' XML for the tracelog section

Usage

```
tree_priors_to_xml_tracelog(tree_priors)
```

Arguments

`tree_priors`	one or more tree priors, as returned by `create_tree_prior`

Author(s)

Richel J.C. Bilderbeek

See Also

the complete tracelog section is created by `create_beast2_input_tracelog`

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

tree_prior_to_xml_operators

Creates the XML of a tree prior, as used in the `operators` *section*

Description

Creates the XML of a tree prior, as used in the `operators` section

Usage

```
tree_prior_to_xml_operators(tree_prior, fixed_crown_age = FALSE)
```

Arguments

`tree_prior` a tree priors, as returned by `create_tree_prior`

`fixed_crown_age`
determines if the phylogeny's crown age is fixed. If FALSE, crown age is estimated by BEAST2. If TRUE, the crown age is fixed to the crown age of the initial phylogeny.

Value

the tree prior as XML text

Author(s)

Richel J.C. Bilderbeek

tree_prior_to_xml_prior_distr

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Description

Creates the distribution section in the prior section of the distribution section of a BEAST2 XML parameter file. These lines start with '<distribution id='

Usage

```
tree_prior_to_xml_prior_distr(tree_prior)
```

Arguments

`tree_prior` a tree priors, as returned by `create_tree_prior`

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

`tree_prior_to_xml_state` *Creates the XML of a tree prior, as used in the `state` section*

Description

Creates the XML of a tree prior, as used in the `state` section

Usage

```
tree_prior_to_xml_state(tree_prior)
```

Arguments

`tree_prior`	a tree priors, as returned by `create_tree_prior`

Value

the tree prior as XML text

Author(s)

Richel J.C. Bilderbeek

tree_prior_to_xml_tracelog

Creates the tree prior's XML for the tracelog section

Description

Creates the tree prior's XML for the tracelog section

Usage

```
tree_prior_to_xml_tracelog(tree_prior)
```

Arguments

tree_prior	a tree priors, as returned by create_tree_prior

Author(s)

Richel J.C. Bilderbeek

See Also

all tree priors' tracelog section is created by tree_priors_to_xml_tracelog

Examples

```
# <logger id="tracelog" ...>
#'   # Here
# </logger>
```

yule_tree_prior_to_xml_prior_distr

Creates the prior *section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Yule tree prior*

Description

Creates the prior section in the prior section of the prior section of the distribution section of a BEAST2 XML parameter file for a Yule tree prior

Usage

```
yule_tree_prior_to_xml_prior_distr(yule_tree_prior)
```

Arguments

`yule_tree_prior`
a Yule tree_prior, as created by `create_yule_tree_prior`

Author(s)

Richel J.C. Bilderbeek

Examples

```
# <distribution id="posterior" spec="util.CompoundDistribution">
#     <distribution id="prior" spec="util.CompoundDistribution">
#       HERE, where the ID of the distribution is 'prior'
#     </distribution>
#     <distribution id="likelihood" ...>
#     </distribution>
# </distribution>
```

Index