Package 'bidsr'

April 2, 2025

Title A Brain Imaging Data Structure ('BIDS') Parser

Version 0.1.0

URL https://dipterix.org/bidsr/, https://github.com/dipterix/bidsr/

BugReports https://github.com/dipterix/bidsr/issues

Description Parse and read the files that comply with the brain imaging data structure, or 'BIDS' format, see the publication from Gorgolewski, K., Auer, T., Calhoun, V. et al. (2016) <doi:10.1038/sdata.2016.44>. Provides query functions to extract and check the 'BIDS' entity information (such as subject, session, task, etc.) from the file paths and suffixes according to the specification. The package is developed and used in the reproducible analysis and visualization of intracranial electroencephalography, or 'RAVE', see Magnotti, J. F., Wang, Z., and Beauchamp, M. S. (2020) <doi:10.1016/j.neuroimage.2020.117341>; see 'citation(``bidsr'')' for details and attributions.

Copyright This software includes material (schema files from directory 'inst/bids-schema') from the 'BIDS' specification, which is licensed under a Creative Commons Attribution 4.0 International License; see a copy of the license under 'inst/bids-schema/LICENSE.txt', and website <https://bids.neuroimaging.io> for details.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

Imports checkmate, data.table, fastmap, fs, jsonlite, nanotime, S7 (>= 0.2.0), utils, uuid

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Zhengjia Wang [aut, cre] (<https://orcid.org/0000-0001-5629-1116>),
Trustees of the University of Pennsylvania [cph] (All files in this
package unless explicitly stated in the file or listed in the
'Copyright' section below.)

Maintainer Zhengjia Wang <dipterix.wang@gmail.com>

Repository CRAN

Date/Publication 2025-04-02 17:40:08 UTC

Contents

`as_bids_tabular` *Class definitions and utilities for 'BIDS' tabular*

Description

Official specification link: https://bids-specification.readthedocs.io/en/stable/common-principles.html#tabular-files. Function `save_tabular` is the high-level generic function that by default calls low-level function `save_bids_tabular_default` by default.

Usage

```
as_bids_tabular(x, ...)

save_bids_tabular(x, path, meta = TRUE, ...)

BIDSTabularColumnDescriptor(..., .list = list())

BIDSTabularMetaSidecar(columns = list())

BIDSTabular(content, meta = NULL)

save_bids_tabular_default(
  x,
  path,
  meta = TRUE,
  compact_meta = TRUE,
  milliseconds = TRUE,
  utc = TRUE,
  ...
)

new_bids_tabular_class(
  table_name,
  parent = BIDSTabular,
  content_setter = NULL,
  meta_preset = NULL,
  prepare_save = NULL,
  lower_case_column_names = FALSE
)
```

Arguments

x	R object that can be converted (e.g. list, table), or a path to a tabular file.
..., .list	for BIDSTabularColumnDescriptor, this is a list of key-value properties; for as_bids_tabular, this is passed to BIDSTabularMetaSidecar
path	path to save the file; the file is always saved as tabular-separated value ('TSV') format
meta	instance of BIDSTabularMetaSidecar, a class containing a list of descriptors for each column (see argument columns)
columns	a named list, where each key correspond to a table column name, and each item is a named list of descriptors, or a BIDSTabularColumnDescriptor instance
content	a data frame or table with column names non-blanks and possibly all in snake-cases (see specification); bidsr does not check on the column names for compatibility concerns. However users should respect the specification and use the recommended conventions
compact_meta	logical, whether the meta side-car ('JSON' file) should use compact format; default is true

milliseconds, utc
: used to convert nanotime to 'BIDS' time-stamp format; default is to keep the milliseconds and use 'UTC' timezone.

table_name
: name of the table, used to generate a new class; the class name will be BIDSTabular_<table_name>

parent
: parent class of the new class; default is BIDSTabular

content_setter
: a setter function to set the content; see bids_property

meta_preset
: a preset function to set the meta; see BIDSTabularMetaSidecar

prepare_save
: a function to prepare the content before saving; should take the BIDSTabular object as the first argument, and return the content to be saved

lower_case_column_names
: if TRUE, the column names will be converted to lower case; default is TRUE

Value

A component in BIDSTabular.

Author(s)

Zhengjia Wang

Examples

```
# convert a data table into BIDS tabular
table <- data.frame(
  a = c(1, 2, 3, NA, NA, 6, 7, 8, 9, 10),
  b = sample(c('a', 'b'), size = 10, replace = TRUE)
)

# basic
as_bids_tabular(table)

# add descriptors
tabular <- as_bids_tabular(
  table,
  a = list(LongName = "An integer"),
  b = list("Levels" = list('a' = "Abnormal", 'b' = "Bipolar"))
)
tabular


# query data
is.data.frame(tabular$content)
tabular$content$a

# query meta
tabular$meta$columns$a

# save to tsv
```

```
tsv <- tempfile(fileext = ".tsv")
paths <- save_bids_tabular(tabular, tsv)
print(paths)

# use base R to read
read.table(tsv, header = TRUE, na.strings = "n/a")

# get sidecar
cat(readLines(paths$sidecar_path), sep = "\n")

unlink(tsv)
unlink(paths$sidecar_path)
```

`BIDSClassBase` *Low-level abstract class for* `bidsr`

Description

Low-level abstract class definition; see `new_bids_class` to create new class definitions. All `bidsr` classes inherit this abstract class, to provide consistent behavior.

Usage

```
BIDSClassBase()
```

Value

Do not call this `S7` class directly, see `new_bids_class` on how to use it properly

Author(s)

Zhengjia Wang

`BIDSDatasetGeneratedBy`

Class definition for 'BIDS' meta-data `'GeneratedBy'`

Description

See definition at https://bids-specification.readthedocs.io/en/stable/glossary.html#objects.metadata.GeneratedBy

Usage

```
BIDSDatasetGeneratedBy(
  Name = character(0),
  Version = character(0),
  Description = character(0),
  CodeURL = character(0),
  Container = list()
)
```

Arguments

`Name`	(character, required) name of the pipeline or process that generated the outputs.
`Version`	(character, optional) version of the pipeline
`Description`	(character, optional) plain-text description of the pipeline or process that generated the outputs.
`CodeURL`	(character, optional) 'URL' where the code used to generate the data may be found.
`Container`	(character, optional) Used to specify the location and relevant attributes of software container image used to produce the data. Valid keys in this object include type, tag 'URL' with string values. Package `'bidsr'` does not check what's inside of this entry.

Value

Instantiated object of class `BIDSDatasetGeneratedBy`

Author(s)

Zhengjia Wang

Examples

```
x <- BIDSDatasetGeneratedBy(
  Name = "RAVE Team",
  Version = "0.0.1",
  Container = list(
    Type = "docker",
    Tag = "rave-ieeg/rave-pipelines:0.0.1"
  )
)

x

x$Version <- "0.0.2"

# convert to basic list
as.list(x)

# get JSON string
```

```
format(x)
```

BIDSEntity *Class definitions of 'BIDS' entity*

Description

A 'BIDS' entity is an attribute that can be associated with a file, contributing to the identification of that file as a component of its file-name in the form of a hyphen-separated key-value pair. The specification can be found at https://bids-specification.readthedocs.io/en/stable/common-principles.html#entities.

Usage

```
BIDSEntity_label_required(
  key = character(0),
  value = character(0),
  index_format = "%d"
)

BIDSEntity_label_optional(
  key = character(0),
  value = character(0),
  index_format = "%d"
)

BIDSEntity_label_prohibited(
  key = character(0),
  value = character(0),
  index_format = "%d"
)

BIDSEntity_index_required(
  key = character(0),
  value = integer(0),
  index_format = "%d"
)

BIDSEntity_index_optional(
  key = character(0),
  value = integer(0),
  index_format = "%d"
)

BIDSEntity_index_prohibited(
  key = character(0),
```

```
  value = integer(0),
  index_format = "%d"
)

BIDSEntity_any_required(
  key = character(0),
  value = character(0),
  index_format = "%d"
)

BIDSEntity_any_optional(
  key = character(0),
  value = character(0),
  index_format = "%d"
)

BIDSEntity_any_prohibited(
  key = character(0),
  value = character(0),
  index_format = "%d"
)
```

Arguments

key	(string, required) A short string, typically a compression of the entity name, which uniquely identifies the entity when part of a file-name.
value	A string (label) or a non-negative integer (index); the requisite form of the value that gets specified alongside the key whenever the entity appears in a file-name. For each entity, the value is of one of two possible types:
index_format	for index entities, how to format index values (e.g. padding zeros) when formatted as string; default is without padding

Index: A non-negative integer, potentially zero-padded for consistent width.

Label: An alphanumeric string. Note that labels must not collide when casing is ignored (`bidsr` does not validate this).

Value

A 'BIDS' entity object.

Author(s)

Zhengjia Wang

Examples

```
entity_int <- BIDSEntity_index_optional(key = "run", value = "001")
entity_int$value <- integer()
```

```
print(entity_int) # nothing will be printed out

# subject entity
entity_subject <- BIDSEntity_any_required(key = "sub", value = "HUP225")

print(entity_subject)

# index
entity_subject$value <- 1

print(entity_subject)

# format index
entity_subject$index_format <- "%03d"
print(entity_subject)


# trying to set invalid values will result in errors
try({
  BIDSEntity_index_required(key = "run")
})


entity_int <- BIDSEntity_index_required(key = "run", value = "001")

# trying to unset require entity
try({
  entity_int$value <- integer()
})

# trying to set invalid entity
try({
  entity_int$value <- "asdad"
})

# trying to set prohibited entiry
try({
  BIDSEntity_index_prohibited("invalid", 123)
})
```

BIDSMap *Low-level nested map to store key-value data with inherited structure*

Description

Low-level nested map to store key-value data with inherited structure

Usage

```
BIDSMap(parent = NULL, search_depth = BIDS_MAP_MAX_DEPTH())
```

Arguments

parent	NULL if the map is at the top level, or another map to inherit
search_depth	integer maximum search depths; default is 29; set options 'bidsr.map.search_depth' or environment variable 'BIDS_MAP_MAX_DEPTH' to change the default depth

Value

A 'BIDSMap' object.

Author(s)

Zhengjia Wang

Examples

```
root_map <- BIDSMap()
root_map$key1 <- 1
root_map$key2 <- 2
names(root_map)

child_map <- BIDSMap(parent = root_map)
child_map$key3 <- 3
names(child_map)
child_map$key1
child_map$key2

# mask key2
child_map$key2 <- "a"
child_map

root_map$key2
child_map$key2

# nested maps
grand_child <- BIDSMap(parent = child_map)

# value comes from child map
grand_child$key2

# remove key2 from child map
child_map@impl$remove("key2")

# key2 is from root map now
grand_child$key2
```

BIDSTabularScans *'BIDS' scans table class*

Description

A tabular containing a list of scans and their metadata. The class is a child class of BIDSTabular, hence see the methods there. The original specification is at https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#scans-file.

Usage

```
BIDSTabularScans(content, meta = NULL)
```

Arguments

content, meta	see BIDSTabular

Value

A BIDSTabularScans instance inheriting BIDSTabular.

Author(s)

Zhengjia Wang

Examples

```
# basic
tabular <- BIDSTabularScans(
  data.frame(
    filename = c(
      "func/sub-control01_task-nback_bold.nii.gz",
      "func/sub-control01_task-motor_bold.nii.gz",
      "meg/sub-control01_task-rest_split-01_meg.nii.gz"
    ),
    acq_time = c(
      "1877-06-15T13:45:30",
      "1877-06-15T13:55:33",
      "1877-06-15T12:15:27"
    )
  )
)

# No ending Z, time is interpreted as local time
# tabular uses UTC time
tabular


# convert existing tabular
```

```
tabular <- BIDSTabular(
  data.frame(
    filename = "func/sub-control01_task-nback_bold.nii.gz",
    acq_time = "1877-06-15T13:45:30"
  )
)
tabular <- as_bids_tabular(tabular, cls = BIDSTabularScans)
tabular

# save to tsv
tsv <- file.path(tempdir(), "scans.tsv")
paths <- save_bids_tabular(tabular, tsv)

print(paths)


# use base R to read
read.table(tsv, header = TRUE, na.strings = "n/a")

# get sidecar
cat(readLines(paths$sidecar_path), sep = "\n")


# clean up
unlink(tsv)
unlink(paths$sidecar_path)
```

`BIDSTabularSessions` *'BIDS' sessions table class*

Description

A tabular containing a list of sessions and their metadata. The class is a child class of `BIDSTabular`, hence see the methods there. The original specification is at https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#sessions-file.

Usage

```
BIDSTabularSessions(content, meta = NULL)
```

Arguments

`content`, `meta`	see `BIDSTabular`

Value

A `BIDSTabularSessions` instance inheriting `BIDSTabular`.

Author(s)

Zhengjia Wang

Examples

```
# basic
tabular <- BIDSTabularSessions(data.frame(
  session_id = c("ses-predrug", "ses-postdrug", "ses-followup"),
  acq_time = c(
    "2009-06-15T13:45:30",
    "2009-06-16T13:45:30",
    "2009-06-17T13:45:30"
  ),
  systolic_blood_pressure = c(120, 100, 110)
))
tabular


# convert existing tabular
tabular <- BIDSTabular(
  data.frame(
    acq_time = "2009-06-15T13:45:30",
    session_id = "ses-predrug",
    systolic_blood_pressure = 120
  )
)
tabular <- as_bids_tabular(tabular, cls = BIDSTabularSessions)
tabular


# save to tsv
tsv <- file.path(tempdir(), "sessions.tsv")
paths <- save_bids_tabular(tabular, tsv)

print(paths)


# use base R to read
read.table(tsv, header = TRUE, na.strings = "n/a")

# get sidecar
cat(readLines(paths$sidecar_path), sep = "\n")


# clean up
unlink(tsv)
unlink(paths$sidecar_path)
```

BIDSURI *'BIDS' uniform resource indicator ('URI') class definition*

Description

'BIDS' uniform resource indicator ('URI') class definition

Usage

```
BIDSURI(uri)
```

Arguments

uri	'URI' string or another 'BIDS-URI' object

Value

A BIDSURI instance.

Author(s)

Zhengjia Wang

Examples

```
# basic properties
uri <- BIDSURI("bids::sub-01/fmap/sub-01_dir-AP_epi.nii.gz")
uri
uri$relative_path
uri$dataset_name

# set the entire uri
uri$format <- "bids:deriv1:sub-02/anat/sub-02_T1w.nii.gz"
uri

# resolve BIDS URI (partial support)


# resolving a BIDS URI requires dataset_description.json
dataset_description <- get_bids_dataset_description(
  parent_directory = "/path/to/BIDS/folder",
  Name = "A dummy experiments",
  BIDSVersion = "1.6.0",

  DatasetLinks = list(
    "deriv1" = "derivatives/derivative1",
    "phantoms" = "file:///data/phantoms"
  )
)
```

```
uri <- BIDSURI("bids::sub-01/fmap/sub-01_dir-AP_epi.nii.gz")
resolved <- resolve_bids_path(uri, dataset_description)

# resolved absolute path
print(resolved)

# `raw_resolution` is relative to the parent directory where
# `dataset_description.json` is stored
attr(resolved, "raw_resolution")

uri <- BIDSURI("bids:deriv1:sub-02/anat/sub-02_T1w.nii.gz")
resolved <- resolve_bids_path(uri, dataset_description)

print(resolved)

attr(resolved, "raw_resolution")
```

`bids_project` *'BIDS' project class*

Description

'BIDS' project class

Usage

```
BIDSProject(
  path,
  raw_data_relpath = ".",
  source_data_relpath = "sourcedata",
  derivative_data_relpath = "derivatives",
  strict = TRUE
)

bids_project(
  path,
  raw_data_relpath = ".",
  source_data_relpath = "sourcedata",
  derivative_data_relpath = "derivatives",
  strict = TRUE
)
```

Arguments

`path`	absolute path to the 'BIDS' project directory;
`raw_data_relpath`	raw data-set path, relative (to the `path`);

`source_data_relpath`
: source data-set path, relative (to the `path`);

`derivative_data_relpath`
: derivative data-set path, relative (to the `path`);

`strict`
: whether `path` needs to exist; default is `TRUE`

Value

A 'BIDS' project instance.

Author(s)

Zhengjia Wang

Examples

```
# Run `download_bids_examples()` first
examples <- download_bids_examples(test = TRUE)
if(!isFALSE(examples)) {

  project_path <- file.path(examples, "ieeg_epilepsy_ecog")

  project <- BIDSProject(
    path = project_path,
    raw_data_relpath = ".",
    derivative_data_relpath = "derivatives"
  )

  project

}
```

---

bids_property *'S7' property for 'BIDS' classes*

Description

Used in `property` to generate properties with constraints in class generators such as `new_bids_class`.

Usage

```
bids_property(
  name,
  class = S7::class_any,
  getter = NULL,
```

```
  setter = NULL,
  validator = NULL,
  default = NULL,
  final = FALSE,
  ...
)

bids_property_optional(
  name,
  class = S7::class_any,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  max_len = 1L,
  ...
)

bids_property_required(
  name,
  class = S7::class_any,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  len = 1L,
  ...
)

bids_property_prohibited(
  name,
  class = S7::class_any,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...
)

bids_property_recommended(
  name,
  class = S7::class_any,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
  max_len = 1L
```

```
)

bids_property_deprecated(
  name,
  class = S7::class_any,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
  max_len = 1L
)

bids_property_character(
  name,
  type = c("optional", "recommended", "required", "deprecated", "prohibited"),
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
  class = S7::class_character
)

bids_property_collapsed_character(
  name,
  type = c("optional", "recommended", "required", "deprecated", "prohibited"),
  collapse = " ",
  ...,
  class = S7::class_character
)

bids_property_choice(
  name,
  choices,
  type = c("optional", "recommended", "required", "deprecated", "prohibited"),
  ...,
  class = S7::class_character
)

bids_property_numeric(
  name,
  type = c("optional", "recommended", "required", "deprecated", "prohibited"),
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
```

```
  class = S7::class_numeric
)

bids_property_integerish(
  name,
  type = c("optional", "recommended", "required", "deprecated", "prohibited"),
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
  class = S7::class_numeric
)

bids_property_list(
  name,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
  class = S7::class_list
)

bids_property_named_list(
  name,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = list(),
  ...,
  class = S7::class_list
)

bids_property_unnamed_list(
  name,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = NULL,
  ...,
  class = S7::class_list
)

bids_property_entity_list(
  name,
  getter = NULL,
  setter = NULL,
```

```
  validator = NULL,
  default = list(),
  ...,
  class = S7::class_list,
  identifier = NULL,
  schema_key = NA,
  bids_version = current_bids_version()
)

bids_property_tabular_column_descriptor_list(
  name,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = list(),
  ...,
  class = S7::class_list
)

bids_property_data_frame(
  name,
  getter = NULL,
  setter = NULL,
  validator = NULL,
  default = data.frame(),
  ...,
  class = S7::class_data.frame
)

bids_property_tabular_content(
  name = "content",
  setter = NULL,
  ...,
  name_meta = "meta",
  lower_case_column_names = FALSE
)

bids_property_tabular_meta(
  name = "meta",
  setter = NULL,
  preset = NULL,
  ...,
  name_content = "content"
)
```

Arguments

name	required, string, name of the property

class	'S7' class
getter, setter, validator, default	see new_property
final	whether the property is final once initialized; default is false; this is for properties that should not be altered
...	passed to other methods
max_len	for type='optional', maximum vector length of the property; default is 1
len	for type='required', vector length of the property; default is 1
type	type of the property, can be 'required', 'optional', or 'prohibited'
collapse	for collapsed property, passed to paste
choices	for properties that can only be chosen from given choices; a character strings of candidate choices.
identifier	"data_type/suffix" combination to get entity rules
schema_key	'BIDS' schema key if explicit entity rules is needed
bids_version	'BIDS' version to query the entity rules
name_meta	for tabular content, the name of the meta property; default is "meta"
lower_case_column_names	for tabular content, whether to convert column names to lower case; default is FALSE
preset	a list of preset meta data; default is NULL
name_content	for tabular meta, the name of the content property; default is "content"

Value

All functions call new_property internally.

Author(s)

Zhengjia Wang

Examples

```
MyClass <- new_bids_class(
  name = "MyClass",
  properties = list(
    str = bids_property_character(
      name = "str",
      type = "required",
      validator = function(value) {
        if (length(value) == 1 &&
            !isTRUE(is.na(value)) && nzchar(value)) {
          return()
        }
        return(sprintf("Invalid `str`: %s", paste(sQuote(value), collapse = ", ")))
      }
```

```
    )
  ),
  methods = list(
    # read-only methods
    format = function(self, ...) {
      sprintf("MyClass@str -> %s", self$str)
    }
  )
)

instance <- MyClass(str = "aha")
instance

instance$str <- "111"
instance


# what if you enter illegal values

try({
  MyClass(str = "")
})

try({
  MyClass(str = NA_character_)
})

try({
  MyClass(str = 1)
})
```

`bids_subject` *'BIDS' subject class*

Description

'BIDS' subject class

Usage

```
BIDSSubject(project, subject_code, ..., strict = "raw")

bids_subject(project, subject_code, ..., strict = "raw")
```

Arguments

project	'BIDS' project instance, see BIDSProject, or a path to the 'BIDS' project
subject_code	character, subject code with or without the leading 'sub-'. The subject code, after trimming the leading entity key, should not contain any additional dash ('-')
...	passed to the constructor of BIDSProject, when project is a character string
strict	whether to check if the subject folders exist, can be logical or characters; when strict is character strings, choices can be 'raw' (checking raw-data directory) and/or 'source' (for source-data directory); strict=TRUE is equivalent to checking both; default is 'raw'. There is no checks on derivatives.

Value

A 'BIDS' subject instance.

Author(s)

Zhengjia Wang

Examples

```
# Run `download_bids_examples()` first
examples <- download_bids_examples(test = TRUE)
if(!isFALSE(examples)) {

  project_path <- file.path(examples, "ieeg_epilepsy_ecog")

  project <- BIDSProject(
    path = project_path,
    raw_data_relpath = ".",
    derivative_data_relpath = "derivatives"
  )


  subject <- BIDSSubject(project = project, subject_code = "ecog01",
                         strict = FALSE)

  storage_root <- resolve_bids_path(subject, storage = "raw")

  query_bids(subject, "ieeg")

}
```

`download_bids_examples` *Download 'BIDS' example data-sets*

Description

See https://github.com/bids-standard/bids-examples for the full repository.

Usage

```
download_bids_examples(test = FALSE)
```

Arguments

`test`	logical; default is `FALSE`, which downloads the example repository if the files are missing; an alternative choice is `TRUE`, which will return `FALSE` if the files are missing

Value

A local path to the example repository exists or when `test=FALSE`; or simply `FALSE` if the repository is missing and `test=TRUE`.

Author(s)

Zhengjia Wang

Examples

```
download_bids_examples(test = TRUE)
```

`get_bids_dataset_description` *Class definition of 'BIDS' data-set description*

Description

See https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#dataset_descriptionjson for specification.

Usage

```
get_bids_dataset_description(x, parent_directory, ...)

BIDSDatasetDescription(
  Name = character(0),
  BIDSVersion = character(0),
  DatasetLinks = list(),
  HEDVersion = character(0),
  DatasetType = character(0),
  License = character(0),
  Authors = character(0),
  GeneratedBy = list(),
  SourceDatasets = list(),
  Acknowledgements = character(0),
  HowToAcknowledge = character(0),
  Funding = character(0),
  EthicsApprovals = character(0),
  ReferencesAndLinks = character(0),
  DatasetDOI = character(0),
  parent_directory = character(0)
)
```

Arguments

x	R object to be interpreted as 'BIDS' data description; default support list, path to the 'json' file, 'json' string, etc.
parent_directory	parent directory where the file 'dataset_description.json' is stored. This input is ignored if x is the path to 'dataset_description.json', otherwise is a must.
...	passed to methods
Name	(required, string) Name of the data-set.
BIDSVersion	(required, string) The version of the BIDS standard that was used.
DatasetLinks	(required if 'BIDS-URI' is used) Used to map a given data-set name from a 'BIDS-URI' of the form bids:<dataset-name>:path/within/dataset to a local or remote location.
HEDVersion	(recommended strings) The version of the 'HED' schema used to validate 'HED' tags for study. May include a single schema or a base schema and one or more library schema.
DatasetType	(recommended string) Must be one of "raw" or "derivative"; package bidsr automatically assigns "raw" is not given.
License	(recommended string) The license for the data-set
Authors	(recommended strings) Vector of individuals who contributed to the creation/curation of the data-set
GeneratedBy	(recommended) will be converted to BIDSDatasetGeneratedBy

`SourceDatasets` Used to specify the locations and relevant attributes of all source data-sets. Valid keys in each object include `"URL"`, `"DOI"`, and `"Version"` with string values; Package `bidsr` does not check the names

`Acknowledgements`
(optional string) Text acknowledging contributions of individuals or institutions beyond those listed in Authors or Funding.

`HowToAcknowledge`
(optional string) Text containing instructions on how researchers using this dataset should acknowledge the original authors. This field can also be used to define a publication that should be cited in publications that use the dataset.

`Funding` (optional strings) List of sources of funding (grant numbers).

`EthicsApprovals`
(optional strings) List of ethics committee approvals of the research protocols and/or protocol identifiers.

`ReferencesAndLinks`
(optional strings) List of references to publications that contain information on the data-set. A reference may be textual or a URI.

`DatasetDOI` (optional string) The Digital Object Identifier of the data-set (not the corresponding paper). `'DOIs'` should be expressed as a valid 'URI'

Value

A `S7` description object that contains all the fields describing the data set; see 'Examples' for usages.

Author(s)

Zhengjia Wang

Examples

```
# ---- Manually enter entries -----------------------------------------
dataset_description <- BIDSDatasetDescription(
  # a parent directory is mandatory as it defines what data
  # dataset_description.json applies to

  parent_directory = "/path/to/BIDS/folder",

  Name = "A dummy experiments",
  BIDSVersion = "1.6.0",
  License = "CC0",
  Authors = c("Zhengjia Wang"),
  Acknowledgements = c(
    "Package `bidsr` is a 3rd-party BIDS reader developed by",
    "a RAVE (https://rave.wiki) team member with procrastination."
  ),
  HowToAcknowledge = c(
    "Please cite this paper:",
    "https://doi.org/10.1016/j.neuroimage.2020.117341"
  ),
```

```
  Funding = c(
    "NIH R01MH133717",
    "NIH U01NS113339",
    "NIH 1R24MH117529"
  ),
  ReferencesAndLinks = c(
    "https://rave.wiki"
  ),
  DatasetDOI = "https://doi.org/10.1016/j.neuroimage.2020.117341",
  HEDVersion = "8.0.0",
  GeneratedBy = list(
    list(
      Name = "Dipterix",
      Version = "0.0.1",
      Container = list(
        Type = "r-package",
        Tag = "dipterix/bidsr:0.0.1"
      )
    )
  )
)

# access the information
dataset_description$License

dataset_description$GeneratedBy[[1]]$Container

# ---- Read from file -------------------------------------------

# Run `download_bids_examples()` first
examples <- download_bids_examples(test = TRUE)
if(!isFALSE(examples)) {
  example_descr <- file.path(
    examples, "ieeg_epilepsy_ecog", "dataset_description.json")

  x <- get_bids_dataset_description(example_descr)
  x

  # ---- Formatting -----------------------------------------------
  # convert to R list (use recursive to expand field `GeneratedBy`)
  as.list(x, recursive = TRUE)

  # JSON string
  format(x)
}
```

---

get_bids_entity *Get 'BIDS' entity values from file*

Description

Get 'BIDS' entity values from file

Usage

```
get_bids_entity(x, key, value_only = TRUE, ifnotfound = NULL)

get_bids_entity_rules(x)
```

Arguments

x	'BIDS' file path or parsed object; see 'Examples'
key	entity key
value_only	whether to return the value only; default is true; set to FALSE to return the entity object
ifnotfound	default value to return is the entity is missing

Value

'BIDS' entity value or object, depending on value_only

Author(s)

Zhengjia Wang

Examples

```
# Quick usage
get_bids_entity("ieeg/sub-YAB_ses-01_task-AV_ieeg.mat", "sub")

get_bids_entity_rules("ieeg/sub-YAB_ses-01_task-AV_channels.tsv")

# Full usage
parsed <- parse_path_bids_entity(
    path = "ieeg/sub-YAB_ses-01_task-AV_channels.tsv")

parsed$get_bids_entity("sub")
parsed$get_bids_entity_rules()

parsed$description
parsed$entities
```

get_bids_participants *'BIDS' participant table class*

Description

A tabular containing a list of participants and their demographics. The class is a child class of BIDSTabular, hence see the methods there. The original specification is at https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#participants-file.

Usage

```
get_bids_participants(x, ...)

BIDSTabularParticipants(content, meta = NULL)
```

Arguments

x	R object such as file path, project instances, etc.
...	passed to other methods or ignored
content, meta	see BIDSTabular

Value

A BIDSTabularParticipants instance inheriting BIDSTabular.

Author(s)

Zhengjia Wang

Examples

```
# basic
tabular <- BIDSTabularParticipants(
  data.frame(
    participant_id = "sub-001"
  )
)
tabular


# Run `download_bids_examples()` first
examples <- download_bids_examples(test = TRUE)
if(!isFALSE(examples)) {

  file <- file.path(examples, "ieeg_epilepsy_ecog", "participants.tsv")
```

```
  # read tabular as BIDSTabularParticipants
  as_bids_tabular(file, cls = BIDSTabularParticipants)

  # convert existing tabular
  tabular <- BIDSTabular(
    data.frame(
      participant_id = "sub-001"
    )
  )
  tabular <- as_bids_tabular(tabular, cls = BIDSTabularParticipants)

  # save to tsv
  tsv <- file.path(tempdir(), "participants.tsv")
  paths <- save_bids_tabular(tabular, tsv)
  print(paths)

  # use base R to read
  read.table(tsv, header = TRUE, na.strings = "n/a")

  # get sidecar
  cat(readLines(paths$sidecar_path), sep = "\n")

  unlink(tsv)
  unlink(paths$sidecar_path)
}
```

`get_bids_phenotype_data` *'BIDS' phenotype and assessment table class*

Description

A tabular containing a list of phenotype & assessment, with their metadata. The class is a child class of BIDSTabular, hence see the methods there. The original specification is at https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#phenotypic-and-assessment-

Usage

```
get_bids_phenotype_data(x, ...)

BIDSTabularPhenotype(content, meta = NULL)
```

Arguments

x	R object such as file path, project instances, etc.
...	passed to other methods or ignored
content, meta	see BIDSTabular

Value

A BIDSTabularPhenotype instance inheriting BIDSTabular.

Author(s)

Zhengjia Wang

Examples

```
BIDSTabularPhenotype(
  meta = list(
    MeasurementToolMetadata = list(
      Description = "Adult ADHD Clinical Diagnostic Scale V1.2",
      TermURL = "https://www.cognitiveatlas.org/task/id/trm_5586ff878155d"
    ),
    adhd_b = list(
      Description = "B. CHILDHOOD ONSET OF ADHD (PRIOR TO AGE 7)",
      Levels = list(
        "1" = "YES",
        "2" = "NO"
      )
    ),
    adhd_c_dx = list(
      Description = "As child met A, B, C, D, E and F diagnostic criteria",
      Levels = list(
        "1" = "YES",
        "2" = "NO"
      )
    )
  ),
  content = data.frame(
    MeasurementToolMetadata = c(2, 3, 4),
    adhd_b = c(1, 2, 1),
    adhd_c_dx = c(2, 1, 2)
  )
)
```

---

get_bids_samples *'BIDS' samples table class*

Description

A tabular containing a list of samples and their metadata. The class is a child class of BIDSTabular, hence see the methods there. The original specification is at https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#samples-file.

Usage

```
get_bids_samples(x, ...)

BIDSTabularSamples(content, meta = NULL)
```

Arguments

x	R object such as file path, project instances, etc.
...	passed to other methods or ignored
content, meta	see BIDSTabular

Value

A BIDSTabularSamples instance inheriting BIDSTabular.

Author(s)

Zhengjia Wang

Examples

```
# basic
tabular <- BIDSTabularSamples(
  data.frame(
    sample_id = "sample-001",
    participant_id = "sub-001",
    sample_type = "cell line"
  )
)
tabular


# convert existing tabular
tabular <- BIDSTabular(
  data.frame(
    sample_id = "sample-001",
    participant_id = "sub-001",
    sample_type = "cell line"
  )
)
tabular <- as_bids_tabular(tabular, cls = BIDSTabularSamples)


# save to tsv
tsv <- file.path(tempdir(), "samples.tsv")
paths <- save_bids_tabular(tabular, tsv)
print(paths)

# use base R to read
read.table(tsv, header = TRUE, na.strings = "n/a")
```

```
# get sidecar
cat(readLines(paths$sidecar_path), sep = "\n")

# clean up
unlink(tsv)
unlink(paths$sidecar_path)
```

`new_bids_class` *Create new* `bidsr` *class definition*

Description

By default, all generated classes inherit `BIDSClassBase`, which provides `S3` generics

Usage

```
new_bids_class(
  name,
  parent = BIDSClassBase,
  abstract = FALSE,
  hidden_names = NULL,
  properties = NULL,
  methods = NULL,
  validator = NULL,
  constructor = NULL
)
```

Arguments

`name`	string, required, name of the class
`parent`	parent class definition, needs to be a `'S7'` class
`abstract`	whether the class is abstract (`TRUE`) or not (`FALSE`)
`hidden_names`	vector of string, names of properties and/or methods whose presence should be hidden from the users; this will affect `` `$` `` operator, or `names` function. The hidden properties or methods cannot be queried via these two ways. However, properties can still be accessible via `` `@` `` operator
`properties`	a named list where the names are the property names that can be queried via `` `$` `` or `` `@` `` operators
`methods`	read-only methods for the class, such as `format` and `print`; if a method is a function, then the arguments should start with `self` (instance method) or `cls` (class method). In most of the cases, changes made to the object will not be carrier out once the the method function exits. For changes to the properties, use `setter` functions in each property.

validator	validate function; see new_class
constructor	function to custom the constructor; see parameter 'constructor' at new_class for details. Basically A custom constructor should call S7::new_object() to create the 'S7' object. The first argument should be an instance of the parent class (if used). The subsequent arguments are used to set the properties.

Value

A S7 object inheriting the 'bidsr::BIDSClassBase' class.

Author(s)

Zhengjia Wang

Examples

```
# ---- Basic usage -------------------------------------------

Range <- new_bids_class(
  "Range",
  properties = list(
    start = bids_property_numeric("start", "required"),
    end = bids_property_numeric("end", "optional")
  ),
  validator = function(self) {
    if(length(self@end) && self@end < self@start) {
      "@end must be great than or equal to @start"
    }
  }
)


r <- Range(start = 10)
r
# get and set properties with @ or $
r$start
r$end <- 40
r$end

try(Range(start = c(10, 15), end = 20))
try(Range(start = 15, end = 10))



# ---- hide properties and attributes -------------------------
MyClass <- new_bids_class(
  name = "MyClass",
  properties = list(
    str = bids_property_character(
      name = "str", type = "required"),
    hidden_prop = bids_property_character("hidden_prop")
  ),
```

```
  methods = list(
    # read-only methods
    format = function(self, ...) {
      sprintf("MyClass@str -> %s", self$str)
    },
    hidden_method = function(self) {
      "Nononononono"
    }
  ),
  hidden_names = c("hidden_method", "hidden_prop")
)

x <- MyClass(str = "a")
x

# hidden names will not be displayed
names(x)
as.list(x)

# however, then can still be queried
x$hidden_prop
x$hidden_method()
```

`new_bids_entity_file_class` *Class generator for 'BIDS' file class with entities*

Description

Low-level function to generate file name definitions with entity constraints; use `parse_path_bids_entity` instead. The specification is at https://bids-specification.readthedocs.io/en/stable/common-principles.html#filenames.

Usage

```
new_bids_entity_file_class(
  name,
  data_type,
  suffix,
  schema_key = NA,
  bids_version = current_bids_version()
)
```

Arguments

name	class name
data_type	'BIDS' file data type
suffix	file suffix
schema_key	schema key if explicit entity rules are required
bids_version	'BIDS' version to query the entity rules

Value

A class definition with proper entity constraints according to data_type-suffix combinations, or a specific schema_key. The function rarely needs to be called directly unless the schema key is missing from the specification.

Author(s)

Zhengjia Wang

Examples

```
# see full table at BIDS specification
# en/stable/appendices/entity-table.html#behavioral-data
#
# generate class definition for "Behavioral Data"
# Entity: Subject Session Task Acquisition Run Recording
# Format:
#   sub-<label> ses-<label> task-<label>
#   acq-<label> run-<index> recording-<label>
# suffix: events
# requirement: REQUIRED OPTIONAL REQUIRED OPTIONAL OPTIONAL
#


# ---- Basic usage ----------------------------------------
behavior_event_file_def <- new_bids_entity_file_class(
  name = "BIDSEntityFile_beh_events",
  data_type = "beh",
  suffix = "events"
)

file1 <- behavior_event_file_def(
  parent_directory = "sub-001/beh",
  sub = "001", task = "test", .extension = "tsv")

print(file1)

file.path("root/to/path", file1)

# How the entities are parsed?
file1$description
```

```
# get entity values
file1$get_bids_entity("task")

# parent directory
file1$parent_directory

file1$entities$run$value

# set entity values
file1$entities$run <- 2
file1$entities$run$index_format <- "%03d"

file1$entities$blahblah <- "haha"

file1

# Relaxed entity rules generated from schema
# `rules.files.raw.task.events` and
# `rules.files.deriv.preprocessed_data.task_events_common`
get_bids_entity_rules(file1)

# ---- Using BIDS schema key for specific version ------------------------
bids_version <- "1.10.1"
behavior_event_file_def <- new_bids_entity_file_class(
  name = "BIDSEntityFile_beh_events",
  data_type = "beh",
  suffix = "events",
  schema_key = "rules.files.raw.task.events",
  bids_version = bids_version
)

file2 <- behavior_event_file_def(
  parent_directory = "sub-001/beh",
  sub = "001", task = "test", .extension = "tsv")

file2$description

# `desc` is no longer listed in the rules here
get_bids_entity_rules(file2)
```

`parse_path_bids_entity` *Parse 'BIDS' entities from file path*

Description

Parse 'BIDS' entities from file path

Usage

```
parse_path_bids_entity(
  path,
  auto_cache = TRUE,
  schema_key = NA,
  bids_version = current_bids_version()
)
```

Arguments

path	path to the entity file, recommended to input the absolute path or relative path from the 'BIDS' root directory
auto_cache	whether to automatically cache the class definition to speed to next time; default is true
schema_key	'BIDS' schema key if explicit entity rules is needed
bids_version	'BIDS' version to query the entity rules

Value

A `'BIDSEntityFile'` instance.

Author(s)

Zhengjia Wang

Examples

```
path <- "anat/sub-01_chunk-001_t1w.nii.gz"

# --- parse ------------------------------------------------
parsed_filename <- parse_path_bids_entity(path)
parsed_filename

parsed_filename$get_bids_entity("sub")

# alternatively
parsed_filename$entities$sub$value

# data type is `anat` imaging
parsed_filename$data_type

# data is T1-weighted
parsed_filename$suffix

# --- usage ------------------------------------------------
# use it as character
file.path("/path/to/bids/dir/sub-01", parsed_filename)

# modify
```

```
parsed_filename$entities$task <- "special"

# new file path: anat/sub-01_task-special_chunk-001_T1w.nii.gz
parsed_filename

# ---- schema -----------------------------------------------
# get BIDS entity rules
parsed_filename$get_bids_entity("task")

# get entity rules
parsed_filename$get_bids_entity_rules()
```

`query_bids` *Query 'BIDS'*

Description

Query 'BIDS' project and analyze the files

Usage

```
query_bids(x, search_params, ...)
```

Arguments

`x`	'BIDS' objects such as subject
`search_params`	searching parameters, leave it blank to see help documentations
`...`	passed to down-stream methods

Value

A data table of query results

Author(s)

Zhengjia Wang

`resolve_bids_path` *Resolve path of a 'BIDS' object*

Description

Resolve path of a 'BIDS' object

Usage

```
resolve_bids_path(x, ...)
```

Arguments

`x`	'BIDS' object such as project or subject
`...`	passed to generic methods

Value

A character of the resolved path

Author(s)

Zhengjia Wang

Examples

```
# ---- BIDS project ------------------------------------------------


# This example needs extra demo files
# Run `download_bids_examples()` first
examples <- download_bids_examples(test = TRUE)

if(!isFALSE(examples)) {

  project_path <- file.path(examples, "ieeg_epilepsy_ecog")

  project <- BIDSProject(
    path = project_path,
    raw_data_relpath = ".",
    derivative_data_relpath = "derivatives"
  )

  # project root
  resolve_bids_path(project, storage = "root")

  # raw-data directory
  resolve_bids_path(project, storage = "raw")
```

```
  # source-data directory
  resolve_bids_path(project, storage = "source")

  # derivatives directory
  resolve_bids_path(project, storage = "derivative")

  # get relative directory to project root
  resolve_bids_path(project, storage = "derivative",
                    relative_to_project = TRUE)

}

# ---- BIDS subject -------------------------------------------------


# This example needs extra demo files
# Run `download_bids_examples()` first
examples <- download_bids_examples(test = TRUE)

if(!isFALSE(examples)) {

  project_path <- file.path(examples, "ieeg_epilepsy_ecog")

  subject <- BIDSSubject(project = project_path,
                         subject_code = "ecog01")

  # raw-data directory
  resolve_bids_path(subject, storage = "raw")

  # source-data directory
  resolve_bids_path(subject, storage = "source")

  # derivatives directory to freesurfer
  resolve_bids_path(subject, storage = "derivative",
                    prefix = "freesurfer")

  # get relative directory to project root
  resolve_bids_path(subject, storage = "raw",
                    relative_to_project = TRUE)

}

# ---- BIDS URI -----------------------------------------------------

# create a BIDS URI
uri <- BIDSURI("bids::sub-01/fmap/sub-01_dir-AP_epi.nii.gz")

# resolving a BIDS URI requires dataset_description.json
data_description <- get_bids_dataset_description(
  parent_directory = "/path/to/BIDS/folder",
  Name = "A dummy experiments",
  BIDSVersion = "1.6.0",
```

```
  DatasetLinks = list(
    "deriv1" = "derivatives/derivative1",
    "phantoms" = "file:///data/phantoms"
  )
)

resolve_bids_path(uri, data_description)
```

Index