Package 'define'

October 15, 2018

Type Package

Title Create FDA-Style Data and Program Definitions

Version 0.2.8

Author Tim Bergsma [aut, cre],
Scott Pivirotto [ctb]

Maintainer Tim Bergsma <bergsmat@gmail.com>

Description Creates a directory of archived files with a descriptive 'PDF' document at the root level (i.e. 'define.pdf') containing tables of definitions of data items and relative-path hyperlinks to the documented files. Converts file extensions to 'txt' per FDA expectations and converts 'CSV' files to 'SAS' Transport format. Relies on data item descriptors stored as per R package 'spec'. See 'package?define'. See also '?define'. Requires a compatible installation of 'pdflatex', e.g. <https://miktex.org/>.

License GPL-3

LazyData TRUE

Imports Hmisc, SASxport, encode, spec, latexpdf

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-10-15 12:20:03 UTC

R topics documented:

`define-package` *Create FDA-style dataset and program definitions.*

Description

define helps you create 'define.pdf' and associated file tree for FDA-style submission of analysis datasets, etc. It converts csv files to SAS Transport V. 5 'xpt' format, using metadata encoded in a specification file. It enforces the 'txt' extension for other (presumably ASCII) files.

Details

The only function you're likely to need from **define** is `define`. You may want to learn more about metadata encoding from **encode** and more about specification files from **spec**.

Author(s)

Tim Bergsma, <bergsmat@gmail.com>

References

FDA Study Data Specifications

FDA PDF Specifications

SAS Transport Format Specification

`as.define` *Coerce to Define*

Description

Coerces to class 'define'. Generic, with method for 'spec'.

Coerces to class 'define' from class 'spec'. Extra arguments ignored.

Usage

```
as.define(x, ...)

## S3 method for class 'spec'
as.define(x, sep = " = ", collapse = "; ",
  escape = character(0), ...)
```

Arguments

x	object
...	passed arguments
sep	separates codes from respective decodes where given
collapse	separates code/decode pairs where given
escape	values to escape for proper latex formatting

Methods (by class)

- spec: method for spec

See Also

specification

as.spec

---

as.document.define *Coerce to Document from Define*

Description

Coerces to class 'document' from class 'define'.

Usage

```
## S3 method for class 'define'
as.document(x, morePreamble = command("usepackage", args =
  "longtable"), geoLeft = "1in", geoRight = "1in", geoTop = "1in",
  geoBottom = "1in", pagestyle = command("pagestyle", args = "plain"), ...)
```

Arguments

x	passed to as.tabular.define
morePreamble	passed to as.document.character
geoLeft	passed to as.document.character
geoRight	passed to as.document.character

geoTop	passed to as.document.character
geoBottom	passed to as.document.character
pagestyle	passed to as.document.character
...	passed to as.document.character and as.tabular.define

---

as.document.submission

Coerce a submission object to a document.

Description

Coerces a submission object to a document.

Usage

```
## S3 method for class 'submission'
as.document(x, title, short = title, protocol = "~",
  sponsor = "~", program = "~", author = "~",
  date = "\\mydate \\today", lhead1 = short, lhead2 = protocol,
  rhead1 = sponsor, rhead2 = program, lfoot = author, rfoot = date,
  logo = NULL, logoscale = 1, morePreamble = NULL, geoLeft = "1in",
  geoRight = "1in", geoTop = "1in", geoBottom = "1in", pagestyle = NULL,
  thispagestyle = NULL, units = FALSE, ...)
```

Arguments

x	a list of artifacts each having attributes: x, tag, des, file, spec
title	a title for the document
short	short title
protocol	relevant protocol
sponsor	program sponsor
program	drug development program
author	document author
date	today's date by default
lhead1	left header 1, e.g. short title
lhead2	left header 2, e.g. relevant protocol(s)
rhead1	right header 1, e.g. sponsor
rhead2	right header 2, e.g. development program
lfoot	left footer (italicized), e.g. responsible party
rfoot	right footer, today's date by default
logo	file path for title page logo
logoscale	size adjustment for logo

morePreamble	if NULL, a special value is passed to as.document.character. See function definition for details, and override if necessary.
geoLeft	passed to as.document.character
geoRight	passed to as.document.character
geoTop	passed to as.document.character
geoBottom	passed to as.document.character
pagestyle	passed to as.document.character
thispagestyle	passed to as.document.character
units	Should units for continuous variables be printed in Codes column?
...	passed to as.define, as.tabular, as.document.character

Details

Makes a pdf-ready character object representing a latex document. Essentially a wrapper for as.document. Title, logo, headers, footers, date are placed on the title page. The second page has a menu (table) of defined objects that creates bi-directional links to any defined data tables. Links are also created to the storage locations relative to the (resulting) define.pdf. Following pages table the attributes of data items in any datasets.

---

as.labeled *Coerce to class labeled.*

Description

Coerces to class labeled.

Coerces character to a labeled data.frame.

Usage

```
as.labeled(x, ...)

## S3 method for class 'character'
as.labeled(x, spec, as.is = TRUE, na.strings = c("",
  "."), rename = function(x, ...) x, ...)
```

Arguments

x	length-one filename for csv-formatted data file
...	passed to as.labeled.dat
spec	length-one file name for spec
as.is	passed to read.csv
na.strings	passed to read.csv
rename	a function with arguments x, ... to pre-process column names

Methods (by class)

- `character`: data.frame method for as.labeled

See Also

`as.labeled.data.frame`

`as.labeled.data.frame` *Coerce data.frame to labeled.*

Description

Coerces data.frame to labeled.

Usage

```
## S3 method for class 'data.frame'
as.labeled(x, label, spec, check = TRUE, ...)
```

Arguments

`x`	data.frame
`label`	a SAS-style label for x
`spec`	a spec (specification) data.frame containing column labels
`check`	should the data.frame be required to match its specification?
`...`	ignored

Details

Positive numeric values less than 1e-70 are coerced to zero to solve SAS encoding issues. Column names are forced unique and forced SAS-compliant with `link{makesasnames}`. Labels are added to the data.frame column names, and to the data.frame itself.

See Also

`as.labeled.character`

as.pdf.define *Coerce to PDF from Define*

Description

Coerces to PDF from class 'define'.

Usage

```
## S3 method for class 'define'
as.pdf(x, stem, ...)
```

Arguments

x	define object
stem	passed to as.pdf
...	passed to as.pdf and as.document

as.submission *Create a submission.*

Description

Returns a list of vetted artifacts (spec or char corresponding to x) that represent a submission object for further processing.

Converts a character vector of file names to class submission.

Usage

```
as.submission(x, ...)

## S3 method for class 'character'
as.submission(x, tag = names(x),
  description = basename(x), dir = ".", subdir = NULL, ...)
```

Arguments

x	filenames: xpt, csv, spec, txt, other
...	passed along to handlers
tag	short names for each element of x
description	informative multi-word label for each element of x
dir	parent directory for placement of submission artifacts
subdir	optional subdirectories relative to dir for each submission artifact

Value

a list of artifacts each having attributes: x, tag, des, file, spec

Methods (by class)

- `character`: character method for as.submission

---

as.submission.submission *Coerce a submission to class submission.*

Description

Coerces a submission to class submission.

Usage

```
## S3 method for class 'submission'
as.submission(x, ...)
```

Arguments

x	object of dispatch
...	passed along

---

as.tabular.define *Coerce to Tabular from Define*

Description

Coerces to class 'tabular' from class 'define'.

Usage

```
## S3 method for class 'define'
as.tabular(x, caption = "", grid = TRUE, rules = 1,
  colwidth = c("1in", "1in", "0.5in", "1.5in", "1.5in"),
  tabularEnvironment = "longtable", walls = 1, tabnum = FALSE,
  pretable = if (is.null(caption)) "" else paste(if (tabnum) "\\caption{"
  else "\\caption*{", caption,
  "}\\\\"), prepos = 1, headerBold = TRUE, reserve = FALSE, ...)
```

Arguments

x	define object
caption	caption for definitions table
grid	passed to as.tabular.data.frame
rules	passed to as.tabular.data.frame
colwidth	passed to as.tabular.data.frame
tabularEnvironment	passed to as.tabular.data.frame
walls	passed to as.tabular.data.frame
tabnum	whether to number the table
pretable	material to include before table, typically a caption
prepos	after which line number should pretable be inserted?
headerBold	whether to use a bold header
reserve	passed to as.tabular
...	passed to as.tabular

---

as.xport *Coerce to class xport.*

Description

Coerces to class xport.

Coerces labeled to xport.

Usage

```
as.xport(x, ...)

## S3 method for class 'labeled'
as.xport(x, name, file, autogen.formats = FALSE, ...)
```

Arguments

x	a labeled data.frame
...	passed to write.xport
name	a name for the data.frame
file	where to write the data.frame
autogen.formats	passed to write.xport

Methods (by class)

- labeled: labeled method for as.xport

define *Define objects per FDA guidance.*

Description

Defines (documents) a set of files in a manner intended to comply with FDA guidance on submission of study data and related documentation. In particular, files in csv format are converted to SAS Transport (xpt), extensions for other files (presumably ASCII) are coerced to txt, files are copied to a directory tree, and define.pdf is created at the top level to describe the files in more detail.

Define a set of files per FDA guidance.

`tag` is taken by default as the names of `x`, but may be supplied explicitly. The following should have the same length as `x`: `tag`, `description`, `subdir`. `subdir` may also be length one. Other arguments have length one.

The function iterates across the elements of `x` to create a 'submission' object, a side effect of which is to copy (conditionally, transformed) each corresponding file to (`subdir` of) `dir`. The 'submission' object is then converted to a pdf, written directly to `dir` as '<stem>.pdf'.

Arguments `short`, `protocol`, `sponsor`, `program`, `author`, and `date` are length-one character that define attributes of the pdf title page. Alternatively, they may be specified by names that reflect position rather than semantics: `lhead1`, `lhead2`, `rhead1`, `rhead2`, `lfoot`, `rfoot` (respectively).

Usage

```
define(x, ...)

## S3 method for class 'character'
define(x, stem = "define", tag = names(x),
  description = basename(x), title = dirname(x[[1]]), short = title,
  protocol = "~", sponsor = "~", program = "~", author = "~",
  date = "\\mydate \\today", logo = NULL, logoscale = 1,
  dir = "./define", subdir = ".", clear = TRUE, units = FALSE, ...)
```

Arguments

`x`	paths to existing files to be documented; possibly a named character vector
`...`	passed to as.submission and as.pdf
`stem`	the base of the file name for the pdf to be created
`tag`	short object names for each element of x; appears in pdf menu, and as table name in XPT file
`description`	informative labels for each element of x
`title`	a title to appear in the pdf
`short`	short title a.k.a. lhead1 (upper left pdf header)
`protocol`	relevant protocols a.k.a. lhead2 (lower left pdf header)
`sponsor`	study sponsor a.k.a. rhead1 (upper right pdf header)

program	drug development program a.k.a. rhead2 (lower right pdf header)
author	document author a.k.a. lfoot left pdf footer, italicized)
date	date format string a.k.a. rfoot (right footer) today's date by default
logo	file path for logo to include on cover page
logoscale	size adjustment for logo
dir	path to directory in which to place pdf and copied (transformed) files
subdir	path to subdirectories to which to copy each (transformed) file represented by x; use NULL to suppress archiving
clear	should dir be deleted if it exists?
units	should units for continuous varables be printed in Codes column?

Value

invisible result of as.pdf. Used for side effects.

Methods (by class)

- character: character method for define

See Also

http://tinyurl.com/fda-pdf-spec-4-0

http://tinyurl.com/fda-study-data-spec-2-0

Examples

```
library(spec)                   # read and write data specifications
library(latexpdf)               # make dummy logo for pdf
library(encode)                 # encode factor levels for spec file

dir <- tempdir()                # a place to experiment
dir <- gsub('\\\\','/',dir)     # clean up windows path
outdir <- file.path(dir,'out')  # where to put the define archive
csv <- file.path(dir,'theoph.csv')    # path to data
script <- file.path(dir,'theoph.R')   # path to script making data
spec <- file.path(dir,'theoph.spec')  # path to data specification

# make dummy logo
## Not run:
as.pdf('{\\huge \\em Pharma, Inc.}',wide = 50, long = 8,stem = 'logo', dir = dir)

## End(Not run)
# browseURL(system.file(package = 'define', 'logo.pdf')) # cached
logo <- system.file(package = 'define', 'logo.pdf')     # path to dummy logo

# make data more interesting
Theoph$renal <- 0
```

```
# create script
code <- "write.csv(x = Theoph,file = csv,row.names = FALSE,quote = FALSE)"
writeLines(code,script)

# 'run' the script
eval(parse(text = code))

# make data specification
s <- specification(Theoph)
renalcat <- c(
 'GFR >= 90 mL/min/1.73m^2',
 '60 <= GFR < 90 mL/min/1.73m^2',
 '45 <= GFR < 60 mL/min/1.73m^2',
 '30 <= GFR < 45 mL/min/1.73m^2',
 'GFR < 30 mL/min/1.73m^2'
)
codes <- encode(0:4, renalcat)
codes
s$guide[s$column == 'renal'] <- codes

write.spec(s,spec)

file.exists(csv)
file.exists(spec)
define(c(theoph = csv),stem = 'minimal',dir = outdir, clean=FALSE)
# browseURL(file.path(outdir,'minimal.pdf'))
# browseURL(system.file(package = 'define', 'minimal.pdf')) # cached

## Not run:
define(
  x = c(
    theodat = csv,
    theoprg = script
   ),
  subdir = c(
    'm5/datasets/analysis/datasets',
    'm5/datasets/analysis/programs'
  ),
  description = c(
    'Theophylline PK Dataset',
    'Theophylline PK Script'
  ),
  title = 'Pharmacokinetics of Theophylline',
  short = 'Theophylline PK',
  protocol = 'Protocol tpk-001',
  sponsor = 'Pharma, Inc.',
  program = 'Theophylline',
  author = 'define package for R',
  logo = logo,
  logoscale = 2,
  clear = FALSE,
  dir = outdir
)
```

```
# browseURL(file.path(outdir,'define.pdf'))

## End(Not run)
# browseURL(system.file(package = 'define','define.pdf')) # cached
# browseURL(system.file(package = 'define','poster.pdf')) # earlier work

# Alternatively, supply aesthetics by position:
## Not run:
define(
  x = c(
    theodat = csv,
    theoprg = script
   ),
  subdir = c(
    'm5/datasets/analysis/datasets',
    'm5/datasets/analysis/programs'
  ),
  description = c(
    'Theophylline PK Dataset',
    'Theophylline PK Script'
  ),
  title = 'Pharmacokinetics of Theophylline',
  lhead1 = 'Theophylline PK',
  lhead2 = 'Protocol tpk-001',
  rhead1 = 'Pharma, Inc.',
  rhead2 = 'Theophylline',
  lfoot = 'define package for R',
  rfoot = '\\mydate \\today',
  logo = logo,
  logoscale = 2,
  clear = FALSE,
  dir = outdir
)

## End(Not run)

# Tags for elements of x can be given explicitly rather than as names:
## Not run:
define(
  x   = c(csv, script),
  tag = c('theodat','theoprg'),
  subdir = c(
    'm5/datasets/analysis/datasets',
    'm5/datasets/analysis/programs'
  ),
  description = c(
    'Theophylline PK Dataset',
    'Theophylline PK Script'
  ),
  title = 'Pharmacokinetics of Theophylline',
  lhead1 = 'Theophylline PK',
  lhead2 = 'Protocol tpk-001',
  rhead1 = 'Pharma, Inc.',
```

```
  rhead2 = 'Theophylline',
  lfoot = 'define package for R',
  rfoot = '\\mydate \\today',
  logo = logo,
  logoscale = 2,
  clear = FALSE,
  dir = outdir
)

## End(Not run)

# If the title is short, no need to supply a short version.
# Most arguments have suitable defaults. But be sure to
# supply tags, or names for elements of x.
## Not run:
define(
  x = c(
    theodat = csv,
    theoprg = script
   ),
  description = c(
    'Theophylline PK Dataset',
    'Theophylline PK Script'
  ),
  title = 'Theophylline PK',
  dir = outdir
)

## End(Not run)
```

`makesasnames` *Make names for a dataset that are unique and follow SAS naming conventions.*

Description

Makes names for a dataset that are unique and follow SAS naming conventions. Modeled after SASxport::makeSasNames, but handling special cases.

Usage

```
makesasnames(names, nchar = 8, maxPasses = 10, quiet = FALSE)
```

Arguments

`names`	existing column names
`nchar`	limit on number of characters

maxPasses	limit on number of reconciliation attempts
quiet	should messages be suppressed?

Value

character

---

recode *Recode an encoded object.*

Description

Recodes an encoded object.

Recodes an encoded character vector.

Usage

```
recode(x, ...)

## S3 method for class 'character'
recode(x, ...)
```

Arguments

x	vector of strings that represent encodings
...	passed to sapply

Details

specification creates a template spec object corresponding to a data frame. Not able to guess factor level decodes (labels) or column descriptions (e.g. SAS labels) it supplies defaults by repeating the argument values. These have value for development, but not for reporting. This function drops non-informative decodes.

Value

character

Methods (by class)

- character: character method for recode

See Also

encode

Index