Package 'DDIwR'

July 30, 2014

Version 0.1-0

Date 2014-07-30

Title DDI related functions

Depends R (>= 3.0.0)

Imports XML, foreign

Description This package provides useful functions for various DDI (Data Documentation Initiative) related outputs.

License GPL (>= 2)

Author Adrian Dusa [aut, cre]

Maintainer Adrian Dusa <dusa.adrian@unibuc.ro>

NeedsCompilation no

Repository CRAN

Date/Publication 2014-07-30 22:02:08

R topics documented:

`DDIwR-package` *Useful functions for various DDI (Data Documentation Initiative) related outputs.*

Description

DDIwR stands for "DDI with R".

This package provides various functions to read DDI based metadata documentation, and write dedicated setup files for R, SPSS, Stata and SAS to read an associated .csv file containing the raw data, apply labels for variables and values and also deal with the treatment of missing values.

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 262608 (DwB - Data without Boundaries)

Details

Package:	DDIwR
Type:	Package
Version:	0.1-0
Date:	2014-07-30
License:	GPL (>= 2)

Author(s)

Authors:
Adrian Dusa
Department of Sociology
University of Bucharest
<dusa.adrian@unibuc.ro>

Maintainer:
Adrian Dusa

`getMetadata` *Get metadata information from an XML file*

Description

This function reads an XML file containing a DDI codebook version 2.5 and returns the variable labels and value labels.

Usage

```
getMetadata(xmlpath, OS = "windows", saveFile = FALSE, ...)
```

Arguments

xmlpath	A path to an XML file with a DDI codebook
OS	The target operating system, for the eol - end of line separator
saveFile	Boolean, save an .R file in the same folder
...	Additional arguments for this function (internal uses only)

Details

The argument OS can be either: "windows" (default), or "Windows", "Win", "win" "MacOS", "Darwin", "Apple", "Mac", "mac" "Linux", "linux"

The end of line separator changes only when the target OS is different from the running OS.

For the moment, only DDI version 2.5 (Codebook) is supported, but the next version will include the latest DDI version 3.2 (Lifecycle).

Value

A list containing two components "varlab" for all variable labels, and "vallab" for all value labels

---

setupfile *Create setup files for SPSS, Stata, SAS and R*

Description

This function creates a setup file, based on a list of variable and value labels.

Usage

```
setupfile(lbls = "", type = "all", csv = "", miss, trymiss = FALSE, uniqueid = "",
          SD = "", delimiter = ",", OS = "windows", outfile = "", ...)
```

Arguments

lbls	The list object containing the variable and value labels as separate components, or a path to the folder where these objects are located, for batch processing.
type	The type of setup file, can be one of: "SPSS", "Stata", "SAS", "R", or "all" (default).
csv	The original dataset, on the basis of which the SPSS setup file commands are created, or a path to the folder where the .csv files are located, for batch processing.
miss	A vector of missing values, or missing labels.

trymiss	Boolean, if TRUE it will try hard to find common missing values (e.g. "DK", or "NA" etc.)
uniqueid	Character, the name of the unique identifier variable
SD	The row delimiter for the Stata commands, can be for example "" or ";"
delimiter	The column delimiter to be used for reading the .csv file, default is ","
OS	The target operating system, for the eol - end of line separator.
outfile	Character, the name of the setup file being created.
...	Other arguments (not used in this function).

Details

If type = "all", it will produce once setup file for each supported type. All created setup files will be saved in a folder called "Setup Files" which (if not already found) will be created in the user's current working directory.

The argument miss expects either: - a vector of missing values (e.g. -1, -2, -3), or - a vector of missing labels

If this is not provided, but trymiss is set to TRUE, then it searches all value labels for these common missing categories: "DK/NA", "DK/NO", "DK", "NA", "N/A", "Not answered", "Don't know", "(Don't know)", "No answer", "No opinion", "Not applicable", "Not relevant", "Refused", "(Refused)", "Refused / no answer", "(Refused / no answer)", "Can't say", "Don't know / Can't say".

If batch processing multiple files, the function will inspect all files in the provided folder, and retain only those with the extension .R or .r or DDI versions with the extension .xml or .XML (it will subsequently generate an error if the .R files do not contain an object list, or if the .xml files do not contain a DDI structured metadata file).

If the metadata folder contains a subfolder called "data" or "Data", it will match the name of the metadata file with the name of the .csv file (their names have to be *exactly* the same, irrespective of their extension).

The csv argument can provide a data.frame object produced by reading the .csv file, or a path to the folder where the .csv files are located. If the user doesn't provide something for this argument, the function will check the existence of a subfolder called data in the folder where the metadata files are located.

The uniqueid argument is only relevant if type = "R". It is necessary to identify missing observations in different variables, based on the unique case identifiers found in the variable provided via this argument. It will generate an "attr"ibute called "missing types", which is essentially a list whose components are variable names, and each component is a list itself containing a vector of values for each missing category (type) plus the identifiers of the cases where missing values are found (and replaced with NA). It will also generate an attribute called "unique id", which points to the same name of the variable containing the unique case identifiers.

The argument SD only makes sense when type = "Stata" or type = "all", (when Stata setup files will also be generated).

In batch mode, the code starts with the argument delimiter = ",", but if the .csv file is delimited differently it will also try hard to find other delimiters which will match the variable names in the metadata file. At the initial version 0.1-0, the automatically detected delimiters include ";" and "\t".

The argument `OS` can be either: `"windows"` (default), or `"Windows"`, `"Win"`, `"win"`, `"MacOS"`, `"Darwin"`, `"Apple"`, `"Mac"`, `"mac"`, `"Linux"`, `"linux"`.

The end of line separator changes only when the target OS is different from the running OS.

The argument `outfile` expects the name of the final setup file being saved on the disk. If nothing is provided, the name of the object provided for the `lbls` argument will be used as a filename.

There is also an undocumented, boolean argument called `saveFile`, which if set to TRUE it will save an R version if the metadata was read from a DDI `.xml` file, in the same folder. This function uses `getMetadata`, where that argument is a formal one.

Value

A setup file to complement the imported raw dataset.

Examples

```
fancyname <- list()

fancyname$varlab <- list(
"ID" = "Questionnaire ID",
"V1" = "First variable's label",
"V2" = "Second variable's label",
"V3" = "Third variable's label"
)


fancyname$vallab$V1 <- c(
"No"             =  0,
"Yes"            =  1,
"Not answered"   = -1
)


fancyname$vallab$V2 <- c(
"Verry little"   =  1,
"Little"         =  2,
"So, so"         =  3,
"Much"           =  4,
"Very much"      =  5,
"Not applicable" = -7,
"Don\'t know"     = -8,
"Not answered"   = -9
)


fancyname$vallab$V3 <- c(
"No"             =  0,
"Yes"            =  1,
"Not answered"   = -1
)
```

```
#
###   IMPORTANT:
##### make sure to set the working directory to a directory with read/write privileges
###
# setwd("/path/to/read/write/directory")


##### then run these commands
###
# setupfile(fancyname, trymiss = TRUE, uniqueid = "ID")


# setupfile(fancyname, type="Stata", SD=";")


##### other types of possible utilizations, using paths to specific files
###
# setupfile("/path/to/the/metadata/file.xml", csv="/path/to/the/csv/file.csv")


##### if the metadata is saved to an .R file containing a list
###
# setupfile("/path/to/the/metadata/file.R", csv="/path/to/the/csv/file.csv")


##### or in batch mode, specifying entire directories
###
# setupfile("/path/to/the/metadata/directory", csv="/path/to/the/csv/directory")
```

Index