Package ‘googlePublicData’

October 13, 2022

Type Package

Title Working with Google's 'Public Data Explorer' DSPL Metadata Files

Version 0.16.1

Date 2017-11-06

Description Provides a collection of functions to set up 'Google Public Data Explorer' <https://www.google.com/publicdata/> data visualization tool with your own data, building automatically the corresponding DataSet Publishing Language file, or DSPL (XML), metadata file jointly with the CSV files. All zip-up and ready to be published in 'Public Data Explorer'.

Depends R (>= 3.2.0)

Imports XML, utils, readxl

License MIT + file LICENSE

URL http://github.com/gvegayon/googlePublicData/

BugReports http://github.com/gvegayon/googlePublicData/issues

LazyLoad yes

RoxygenNote 6.0.1

Suggests testthat, covr, googleVis

NeedsCompilation no

Author George Vega Yon [aut, cre]

Maintainer George Vega Yon <g.vegayon@gmail.com>

Repository CRAN

Date/Publication 2017-11-06 22:06:57 UTC

R topics documented:

`checkTimeFormat` *DSPL time format verification*

Description

Checks if a string fulfills the joda-times class specifications supported by DSPL language.

Usage

```
checkTimeFormat(fmt)
```

Arguments

`fmt` String representing a time format to be verified.

Details

Public Data Explorer currently supports daily, monthly and yearly distributed data. Joda-time, the corresponding time format on which DSPL times is based, allows declaring time formats using small case "d" (for days), capitalized "M" (for months) and small case "y" for years. Some examples:

Format Specification	Data Example
"yyyy"	1988
"yyyy-MM"	1988-03
"yyyy-MMM"	1988-Mar
"dd-MM-yyyy"	02-03-1988

Value

Logical. `TRUE` if the string passes the test.

Author(s)

George G. Vega Yon

References

- Google Public Data Explorer DSPL time definition: https://developers.google.com/public-data/docs/canonical/time?hl=es
- Google Public Data Explorer Cookbook for time definitions: https://developers.google.com/public-data/docs/cookbook#time_recipes
- Joda Time 2.1 API: http://joda-time.sourceforge.net/api-release/org/joda/time/format/DateTimeFormat.html

See Also

See also dspl

Examples

```
checkTimeFormat("yyyy-MM") # TRUE
checkTimeFormat("MMMyyyy") # TRUE
checkTimeFormat("mmmyyyy") # FALSE
```

countries   *World countries example data set*

Description

This data set is one used in the DSPL Tutorial. Specifically, it contains the basic columns used to define geographical dimensions, in this case, countries.

Format

A data frame containing 5 observations.

Source

DSPL Google Code Page Downloads: https://developers.google.com/public-data/docs/tutorial

country_slice *Some countries statistics*

Description

This data set is one used in the DSPL Tutorial. Specifically, it contains the population magnitudes at country level since 1960 to 1963.

Format

A data frame containing 13 observations.

Source

DSPL Google Code Page Downloads: https://developers.google.com/public-data/docs/tutorial

dspl *Builds Dataset Publication Language (DSPL) metadata file*

Description

Parsing *csv*, *tab* or *xls(x)* files at a specific directory path, dspl generates a complete DSPL file. If an output string is specified, the function generates the complete ZIP (DSPL file plus csv files) ready to be uploaded to Google Public Data Explorer.

Usage

```
dspl(path, output = NA, replace = F, targetNamespace = "",
  timeFormat = "yyyy", lang = c("es", "en"), name = NA,
  description = NA, url = NA, providerName = NA, providerURL = NA,
  sep = ";", dec = ".", encoding = getOption("encoding"),
  moreinfo = NULL)

new_dspl(path, output = NA, replace = F, targetNamespace = "",
  timeFormat = "yyyy", lang = c("es", "en"), name = NA,
  description = NA, url = NA, providerName = NA, providerURL = NA,
  sep = ";", dec = ".", encoding = getOption("encoding"),
  moreinfo = NULL)
```

Arguments

`path` String. Path to the folder where the tables (csv|tab|xls) are at.

`output` String, optional. Path to the output ZIP file.

`replace` Logical. If `output` ZIP file is defined exists, `dspl` replaces it.

`targetNamespace` String. As DSPL documentation states “Provides a URI that identifies your dataset. This URI is not required to point to an actual resource, but it’s a good idea to have the URI resolve to a document describing your content or dataset”.

`timeFormat` String. The corresponding time format of the collection. Should be specified accordingly to joda-time format. See the Details section for more information.

`lang` A list of strings of the languages supported by the dataset. Could be only one.

`name` List of strings. The name of the dataset as defined accordingly to the `lang` list.

`description` List of strings. Description of the dataset. It also supports multiple description as the `name`

`url` The corresponding URL for the dataset.

`providerName` List of strings. The data provider name.

`providerURL` List of strings. The data provider website url.

`sep` The separation character of the tables in the ’path’ folder. Currently supports introducing the following arguments: “,” or “;” (for .csv files), “\t” (for .tab files) and “xls” or “xlsx” (for Microsoft’s excel files).

`dec` String. Decimal point.

`encoding` The char encoding of the input tables. Currently ignored for Microsoft excel files.

`moreinfo` A special tab file generated by the function `genMoreInfo` that contains a dataframe of the dataset concepts with more specifications such as description, topic, url, etc.

Details

If there isn’t any output defined the function returns a list of class `dspl` that among its contents has a xml object (DSPL file); otherwise, if an output is defined, the results consists on two things, an already ZIP file containing a all the necessary to be uploaded at `publicdata.google.com` (a collection of csv files and the XML DSPL written file) and a message (character object).

Internally, the parsing process consists on the following steps:

1. Loading the data,
2. Generating each column corresponding id,
3. Identifying the data types,
4. Building concepts,
5. Identifying dimensional concepts and distinguishing between categorical, geographical and time dimensions, and
6. Executing internal checks.

In order to properly load the zip file (DSPL file plus CSV data files), the function executes a series of internal checks upon the data structure. The detailed list:

- **Slices with the same dimensions**: DSPL requires that each slice represents one dimensional cut, this is, there should not be more than one data table with the same dimensions.
- **Duplicated concepts**: As a result of multiple data types, e.g a single concept (statistic) as integer in one table and float in other, `dspl` may get confused, so during the parsing process, if there is a chance, it collapses duplicated concepts into only one concept and assigns it the common data type (float).
- **Correct time format definition**: Using `checkTimeFormat` ensures that the time format specified is compatible with DSPL.

Value

If there isn't any `output` defined, `dspl` returns list of `class` `"dspl"`.

An object of class `"dspl"` is a list containing:

`dspl` A character string containing the DSPL XML document as defined by the `saveXML` function.

`concepts.by.table` A data frame object of concepts stored by table.

`dimtabs` A data frame containing dimensional tables.

`slices` A data frame of slices.

`concepts` A data frame of concepts (all of them).

`dimensions` A data frame of dimensional concepts.

`statistics` A matrix of statistics.

otherwise the function will build a ZIP file as specified in the output containing the CSV and DSPL (XML) files.

Author(s)

George G. Vega Yon

References

- Google Public Data Explorer Tutorial: `https://developers.google.com/public-data/docs/tutorial`

Examples

```
demo(dspl)
```

dspl-methods *Print and summarize dspl objects*

Description

Methods to print and summarize `dspl` class objects

Usage

```
## S3 method for class 'dspl'
print(x, path = NULL, replace = FALSE, quiet = FALSE, ...)

## S3 method for class 'dspl'
summary(object, ...)
```

Arguments

`x`	An object of class `dspl` to be printed.
`path`	String. Output path where to save the XML DSPL file.
`replace`	Logical. If `path` exists, `TRUE` would replace the file.
`quiet`	Whether or not to print information on the screen
`...`	arguments passed on to `cat` (`print.dspl`)
`object`	An object of class `dspl` to be summarized.

Value

`list("print.dspl")`
: None (invisible `NULL`).

`list("summary.dspl")`
: Returns the class attributes and a list containing as defined by `dspl` function. For more information see its value section.

Author(s)

George G. Vega Yon

See Also

See also `dspl`

Examples

```
  ## Not run:
    # Parsing some xlsx files at "my stats folder"
    mydspl <- dspl(path="my stats folder/")

    # Checking the summary of the data bundle
    summary(mydspl)

    # Writing the DSPL XML definition into a file
    outputfile <- tempfile()
    print(mydspl, path=outputfile)


## End(Not run)
```

`genders` *Genders example data set*

Description

This data set is one used in the DSPL Tutorial. Specifically, it contains the basic columns used to define a categorical dimensions such as gender.

Format

A data frame containing 2 observations.

Source

DSPL Google Code Page Downloads: https://developers.google.com/public-data/docs/tutorial

---

`gender_country_slice` *Some Countries statistics at Gender level*

Description

This data set is one used in the DSPL Tutorial. Specifically, it contains the population magnitudes at country and gender level since 1960 to 1961.

Format

A data frame containing 13 observations.

Source

DSPL Google Code Page Downloads: https://developers.google.com/public-data/docs/tutorial

genMoreInfo *Generates a dataframe used to complement a DSPL bundle*

Description

Parsing *csv*, *tab* or *xls(x)* files at a specific directory path, genMore info generates a dataframe used to complete a DSPL bundle with a more complete concepts definition including description, url, etc..

Usage

```
genMoreInfo(path, encoding = getOption("encoding"), sep = ";",
  output = NA, action = "merge", dec = ".")
```

Arguments

`path`	String. Path to the folder where the tables are saved.
`encoding`	The encoding of the files to be parsed.
`sep`	The separation character of the tables in the 'path' folder. Currently supports introducing the following arguments: "," or ";" (for .csv files), "\t" (for .tab files) and "xls" or "xlsx" (for Microsoft's excel files).
`output`	If defined, the place where to save the dataframe as tab file. Otherwise it returns a data frame object.
`action`	Tells the function what to do if there's a copy of the file. Available actions are "merge" and "replace".
`dec`	String. Decimal point.

Details

If there isn't any output defined (`NA`) the function returns a dataframe containing concepts as observations. Using this, the user may add more descriptive info about concepts. In turn it writes a tab file with the dataframe described above. The user may recycle this file writing "append" in the `action` argument.

Value

If no `output` defined, `genMoreInfo` returns a dataframe with the following columns.

`id`	XML id of the concept (autogenerated)
`label`	The label of the concept (autogenerated)
`description`	A brief description of the concept
`topic`	The topic of the concept
`url`	A URL for the concept where, for example, to get more info
`totalName`	A total name as specified by DSPL language (works for dimensional concepts)
`pluralName`	A total name as specified by DSPL language (works for dimensional concepts)

Author(s)

George G. Vega Yon

References

Google Public Data Explorer: http://publicdata.google.com

Examples

```
# Getting the path where all the datasets are
path <- system.file("dspl-tutorial", package="googlePublicData")
info <- genMoreInfo(path) # This is a dataframe

# Setting the 5th concept as topic "Demographics"
info[5, "topic"] <- "Demographics"

# Generating the dspl file
ans <- dspl(path, moreinfo = info)
ans

  ## Not run:
    # Parsing some xlsx files at "my stats folder" to gen a "moreinfo" dataframe
    INFO <- genMoreInfo(path="my stats folder/", sep="xls")

    # Rows 1 to 10 are about "Poverty" and rows 11 to 20 about "Education"
    # So we fill the "topic" column with it.
    INFO$topic[1:10] <- "Poverty"
    INFO$topic[11:20] <- "Education"

    # Finally, we build the DSPL ZIP including more info
    dspl(path="my stats folder/", sep="xls", moreinfo=INFO)


## End(Not run)
```

googlePublicData *Working with Google's Public Data Explorer DSPL Metadata Files*

Description

googlePublicData package provides a collection of functions to set up Google Public Data Explorer data visualization tool with your own data, building automatically the corresponding DSPL (XML) metadata file jointly with the CSV files. All zipped up and ready to be published at Public Data Explorer.

Details

Also includes several data structure verifiers in order to avoid surprises while loading your ZIP file to Public Data Explorer page.

Please visit the project home for more information and examples: http://github.com/gvegayon/googlePublicData.

Author(s)

George G. Vega Yon

References

- googlePublicData project site: http://github.com/gvegayon/googlePublicData
- Public Data Explorer site: http://publicdata.google.com/
- Public Data Explorer Developers site: https://developers.google.com/public-data/
- googleVis package: https://github.com/mages/googleVis#googlevis

Examples

```
  ## Not run:
    demo(dspl)

## End(Not run)
```

---

states *US states example data set*

Description

This data set is one used in the DSPL Tutorial. Specifically, it contains the basic columns used to define geographical dimensions, in this case, US States.

Format

A data frame containing 8 observations.

Source

DSPL Google Code Page Downloads: https://developers.google.com/public-data/docs/tutorial

state_slice *Some US States statistics*

Description

This data set is one used in the DSPL Tutorial. Specifically, it contains the population magnitudes and unemployment rate at state level since 1960 to 1963.

Format

A data frame containing 9 observations.

Source

DSPL Google Code Page Downloads: https://developers.google.com/public-data/docs/tutorial

Index