

Package ‘pbixr’

October 14, 2022

Type Package

Title Access Data and Metadata from 'Microsoft' 'Power BI' Documents

Version 0.1.4

Author Don Diproto [aut, cre]

Maintainer Don Diproto <package.pbixr@gmail.com>

Description Access data and metadata from 'Microsoft' 'Power BI' ('.pbix', <<https://powerbi.microsoft.com>>) documents with R. The 'pbixr' package enables one to extract 'Power Query M' formulas (<<https://docs.microsoft.com/en-us/power-query/>>) 'Data Analysis Expressions' queries ('DAX', <<https://docs.microsoft.com/en-us/dax/>>) and their properties, report layout and style, and data and data models.

URL <https://github.com/pbixr/pbixr>

BugReports <https://github.com/pbixr/pbixr/issues>

Depends R (>= 3.2.0), dplyr

License GPL-3

Encoding UTF-8

LazyData true

Imports formatR, xml2, jsonlite, zip, utils, textclean, stringr

Suggests knitr, rmarkdown, testthat (>= 2.1.0), RCurl, ggplot2, ggraph, igraph, imager, tidyr

VignetteBuilder knitr

RoxygenNote 7.0.0

SystemRequirements 'Microsoft' 'PowerShell' (<<https://docs.microsoft.com/en-us/powershell/>>), 'Microsoft' 'Power BI' (<<https://powerbi.microsoft.com>>)

NeedsCompilation no

Repository CRAN

Date/Publication 2020-10-27 10:50:03 UTC

R topics documented:

f_clean_under_the_hood	2
f_compress_pbix	4
f_decompress_pbix	5
f_extract_images	6
f_get_connections	8
f_get_dama	9
f_get_dama_file	10
f_get_dama_index	12
f_get_dama_m	13
f_get_dama_xml	14
f_get_dama_xml_data	16
f_get_dama_xml_details	17
f_get_pbix_fir	18
f_get_pbix_info	20
f_query_datamodel	21
f_read_any_json	23
f_read_layout	25
f_remove_file	26
f_search_xml	28

Index	30
--------------	-----------

f_clean_under_the_hood

Remove 'DataModel' from the Collection of Files Compressed in a '.pbix'

Description

'pbix' is decompressed, making its collection of files available for manipulation. 'DataModel' is removed from the collection. Files remaining in the collection are (1) compressed to form a modified 'pbix' and (2) deleted.

Usage

```
f_clean_under_the_hood(input_file_pbix, collection_files_pbix, output_pbix)
```

Arguments

input_file_pbix Path of the input '.pbix'.

collection_files_pbix Directory of the decompressed files associated with the '.pbix'.

output_pbix Path of the modified '.pbix'.

Value

None

Author(s)

Don Diproto

See Also

Uses: [f_remove_file](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

pathFileSampleMod <- file.path(temp_dir, "sample_modified_f10.pbix")
dirFileSampleMod <- file.path(temp_dir, "sample_modified_f10")
# Remove output file and directory if they exist
if(file.exists(pathFileSampleMod)) {
  file.remove(pathFileSampleMod)
}
if(dir.exists(dirFileSampleMod)) {
  unlink(dirFileSampleMod, recursive = TRUE)
}
# Run the function
f_clean_under_the_hood(pathFileSample, dirFileSampleMod, pathFileSampleMod)

## End(Not run)
```

f_compress_pbix	<i>Convert a Collection of Files to a '.pbix'</i>
-----------------	---

Description

A collection of files from, or similar in structure to, a decompressed '.pbix' is compressed, generating a '.pbix'.

Usage

```
f_compress_pbix(collection_files_pbix, output_pbix)
```

Arguments

```
collection_files_pbix
    Directory of the collection of files.
output_pbix          Path of the modified '.pbix'.
```

Value

None

Author(s)

Don Diproto

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
}
```

```
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

pathFileSampleMod2 <- file.path(temp_dir, "sample_modified_f30.pbix")
dirFileSampleMod2 <- file.path(temp_dir, "sample_modified2_f30")
if(file.exists(pathFileSampleMod2) ) {
  file.remove(pathFileSampleMod2)
}
if(dir.exists(dirFileSampleMod2)) {
  unlink(dirFileSampleMod2, recursive = TRUE)
}
# Run the function
f_compress_pbix(dirFileSampleMod2, pathFileSampleMod2)

## End(Not run)
```

f_decompress_pbix *Decompress '.pbix' to a Collection of Files*

Description

A '.pbix' is decompressed, making its collection of files available for manipulation.

Usage

```
f_decompress_pbix(input_file_pbix, collection_files_pbix)
```

Arguments

input_file_pbix
 Path of the input '.pbix'.

collection_files_pbix
 Directory of the collection of files.

Value

None

Author(s)

Don Diprto

Examples

```

## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

output_pbix_file <- gsub("OR_sample_func.pbix", "OR_unzip_pbix",
  pathFileSample)
# Run the function
f_decompress_pbix(pathFileSample, output_pbix_file)

## End(Not run)

```

f_extract_images

Get an Image File from the Collection of Files Compressed in a '.pbix'

Description

The collection of files compressed in a '.pbix' is searched for images. An image is written to a temporary file. The path of the temporary file and associated properties are returned.

Usage

```
f_extract_images(input_file_pbix, image_reg)
```

Arguments

```
input_file_pbix
  Path of the input '.pbix'.
```

image_reg Pattern used to search for an image file stored in a collection of files compressed in '.pbix' (e.g., "[.png|.jpg]").

Value

A list: [[1]] a temporary location for an image, [[2]] the name, length (kb) and date associated with an image.

Author(s)

Don Diproto

See Also

Uses: [f_get_pbix_info](#), [f_get_pbix_fir](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

image_reg <- "[.png|.jpg]"
# Run the function
test <- f_extract_images(pathFileSample, image_reg)
# Plot the image
im <- imager::load.image(test[[1]])
plot(im)

## End(Not run)
```

f_get_connections *Get 'Analysis Services' Connections to an Open '.pbix'*

Description

A query to link an open '.pbix'(s) with relevant 'Analysis Services' connection information is developed. The query is executed via 'PowerShell'.

Usage

```
f_get_connections()
```

Value

The '.pbix' and associated port.

Note

An input is not required for this function. 'Power BI' and 'PowerShell' are required.

Author(s)

Don Diproto

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
```



```

}
# Do stuff -----

#
# Open the .pbix with 'Power BI' if it is not already open.
#
# Run the function
connections_open <- f_get_connections()

## End(Not run)

```

f_get_dama	<i>Get 'DataMashup' from the Collection of Files Compressed in a '.pbix'</i>
------------	--

Description

The byte sequence of 'DataMashup' within a '.pbix' is retrieved.

Usage

```
f_get_dama(input_file_pbix)
```

Arguments

```
input_file_pbix
    Path of the input '.pbix'.
```

Value

'DataMashup' within a '.pbix'.

Author(s)

Don Diproto

See Also

Uses: [f_get_pbix_fir](#).

Examples

```

## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"

```

```

pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Run the function
test <- f_get_dama(pathFileSample)

## End(Not run)

```

f_get_dama_file *Get a File within 'DataMashup'*

Description

The byte sequence of 'DataMashup' within a '.pbix' is retrieved and the relevant file within 'DataMashup' is extracted.

Usage

```
f_get_dama_file(input_file_pbix, variable, index_collection)
```

Arguments

input_file_pbix
Path of the input '.pbix'.

variable
File to be extracted ("xml", "zip", "ziponly" or "hf"). "xml" refers to one or more uncompressed '.xml' files inside 'DataMashup'. "zip" refers to compressed (.zip) data within 'DataMashup'. "ziponly" refers to "zip" excluding "xml". "hf" refers to data occurring before and after compressed data.

index_collection
Index created with f_get_dama_index.

Value

The byte sequence of 'DataMashup' based on an index.

Author(s)

Don Diproto

See AlsoUses: [f_get_dama](#).**Examples**

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

index_collection <- f_get_dama_index(pathFileSample)
# Run the function with different options
# xml
test_xml <- f_get_dama_file(pathFileSample, "xml", index_collection)
# zip
test_zip <- f_get_dama_file(pathFileSample, "zip", index_collection)
# ziponly
test_zip <- f_get_dama_file(pathFileSample, "ziponly", index_collection)
# hf
test_hf <- f_get_dama_file(pathFileSample, "hf", index_collection)

## End(Not run)
```

f_get_dama_index *Get Byte Index of Files within 'DataMashup'*

Description

The position of the start and end bytes of different files, or their components, within 'DataMashup' are identified.

Usage

```
f_get_dama_index(input_file_pbix)
```

Arguments

input_file_pbix
 Path of the input '.pbix'.

Value

Bytes of 'DataMashup', including [[1]] start and end of a parsable '.zip' file, [[2]] start of each '.zip' signature, [[3]] start and end of xml, and [[4]] total length of 'DataMashup'.

Author(s)

Don Diproto

See Also

Uses: [f_get_dama](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
```

```

url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
url <- paste0(url_pt1, url_pt2)
req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Run the function
test <- f_get_dama_index(pathFileSample)

## End(Not run)

```

f_get_dama_m

Get 'Power Query M' Formula within 'DataMashup'

Description

The byte sequence of 'DataMashup' within '.pbix' is retrieved and the 'Power Query M' formula is extracted.

Usage

```
f_get_dama_m(input_file_pbix, remove_temp)
```

Arguments

input_file_pbix Path of the input '.pbix'.

remove_temp Option to remove temporary zip file.

Value

'Power Query M' formula.

Author(s)

Don Diproto

See Also

Uses: [f_get_dama_index](#), [f_get_dama_file](#).

Examples

```

## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Run the function
test <- f_get_dama_m(pathFileSample, TRUE)

## End(Not run)

```

f_get_dama_xml	<i>Get '.xml' within 'DataMashup'</i>
----------------	---------------------------------------

Description

The byte sequence of 'DataMashup' within a '.pbix' is retrieved and the '.xml' is extracted.

Usage

```
f_get_dama_xml(input_file_pbix, xml_start, xml_end)
```

Arguments

input_file_pbix	Path of the input '.pbix'.
xml_start	Start position of '.xml'
xml_end	End position of '.xml'

Value

The '.xml' Within 'DataMashup'.

Author(s)

Don Diproto

See Also

Uses: [f_get_dama_index](#), [f_get_dama_file](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Get the start and end positions
test <- f_get_dama_xml_details(pathFileSample)
xml_start <- (test[[1]][1]/2) + 1
xml_end <- test[[3]][1]
# Run the function
output <- f_get_dama_xml(pathFileSample, xml_start, xml_end)

## End(Not run)
```

f_get_dama_xml_data *Get Data from an '.xml' within 'DataMashup'*

Description

The '.xml' extracted from 'DataMashup' is queried.

Usage

```
f_get_dama_xml_data(input_file_xml)
```

Arguments

input_file_xml The '.xml' within 'DataMashup'.

Value

Data from the '.xml' within 'DataMashup'.

Author(s)

Don Diproto

See Also

Uses: [f_search_xml](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
```



```

    req <- download.file(url, destfile = pathFileSample, mode = "wb")
  } else {
    pathFileSample <- existing_file[1]
  }
  # Do stuff -----

  # Get the start and end positions
  test <- f_get_dama_xml_details(pathFileSample)
  xml_start <- (test[[1]][1]/2) + 1
  xml_end <- test[[3]][1]
  # Get the .xml Within DataMashup
  output <- f_get_dama_xml(pathFileSample, xml_start, xml_end)
  # Run the function
  get_xml_data <- f_get_dama_xml_data(output)

  ## End(Not run)

```

f_get_dama_xml_details

Get Details of an '.xml' within 'DataMashup'

Description

The details of an '.xml' within 'DataMashup' are retrieved.

Usage

```
f_get_dama_xml_details(input_file_pbix)
```

Arguments

input_file_pbix
Path of the input '.pbix'.

Value

A list containing [[1]] the length of each '.xml', [[2]] the first 400 bytes of each '.xml' converted to character and [[3]] the total length of all '.xml' files.

Author(s)

Don Diproto

See Also

Uses: [f_get_dama_index](#), [f_get_dama_file](#),

Examples

```

## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Run the function
test <- f_get_dama_xml_details(pathFileSample)

## End(Not run)

```

f_get_pbix_fir

Get the Byte Sequence of a File from the Collection of Files Compressed in a '.pbix'

Description

'pbix' is decompressed in memory, making its collection of files available for manipulation. The byte sequence of a specific file in the collection is retained. Files in the collection can be identified with f_get_pbix_info.

Usage

```
f_get_pbix_fir(input_file_pbix, variable)
```

Arguments

input_file_pbix Path of the input '.pbix'.

variable Name of file in the collection of files.

Value

Byte sequence of a file.

Note

f_get_pbix_fir included modification of a function ('zip_buffer') from the 'readxl' package (licence GPL-3). The function could not be imported from readxl at the time of 'pbix' publication. 'zip_buffer' was available from: <https://github.com/tidyverse/readxl/blob/master/R/xlsx-zip.R>.

Author(s)

Don Diproto

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

variable <- paste0("Report/CustomVisuals/ImgViewerVisual1455487926945/",
  "resources/ImgViewerVisual.css")
# Run the function
```

```
test <- f_get_pbix_fir(pathFileSample, variable)

## End(Not run)
```

f_get_pbix_info *Identify Collection of Files Compressed in '.pbix'*

Description

'.pbix' is decompressed in memory, making names and properties (length, date) of files in collection available.

Usage

```
f_get_pbix_info(input_file_pbix)
```

Arguments

input_file_pbix
Path of the input ".pbix".

Value

data.frame: Names, lengths (kb) and dates associated with collection of files in '.pbix'.

Author(s)

Don Diproto

Examples

```
## Not run:
# Create a temporary directory
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
```

```

url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
url <- paste0(url_pt1, url_pt2)
req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Run the function
test <- f_get_pbix_info(pathFileSample)

## End(Not run)

```

f_query_datamodel *Query 'DataModel' of a '.pbix'*

Description

A query of 'DataModel' of a '.pbix' currently open in 'Power BI' is developed. The query is exchanged with 'Analysis Services' via 'PowerShell'. Results are written to a temporary file, which is (1) read into R and (2) deleted.

Usage

```
f_query_datamodel(queryPowerBI, connection_string)
```

Arguments

queryPowerBI Query of 'DataModel' (e.g. 'DAX', 'MDX').

connection_string Connection to 'DataModel' initiated in 'Analysis Services'. Please note: (1) '.pbix' must be open in 'Power BI' to connect to 'DataModel' and (2) the identifier and port used in the connection change each time a '.pbix' is opened with 'Power BI'.

Value

Result from a query of 'DataModel'. For one table, a data.frame is returned. For many tables, a list is returned. For an error, perhaps due to incorrect 'DAX' or 'MDX' or incorrect connection, a list of 1 equal to NULL.

Note

'Power BI' and 'PowerShell' are required.

Author(s)

Don Diprto

Examples

```

## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----
OR_pathFileSample <- pathFileSample

# Open the .pbix with 'Power BI' if it is not already open.
#
# Identify the right port
connections_open <- f_get_connections()
connections_open$pbix <- gsub(" - Power BI Desktop", "",
  connections_open$pbix_name)
connections_open <- connections_open[which(connections_open$pbix ==
  gsub("[.]pbix", "", basename(OR_pathFileSample))), ][1, ]
correct_port <- as.numeric(connections_open$ports)
# Construct the connection
connection_db <- paste0("Provider=MSOLAP.8;Data Source=localhost:",
  correct_port, ";MDX Compatibility=1")

# Example 1
# No need to change the syntax
queryPowerBI <- "evaluate TopMovies"
getQueryPowerBIData <- f_query_datamodel(queryPowerBI, connection_db)
str(getQueryPowerBIData)

# Example 2
# Escape dollar sign so that it can run via PowerShell
queryPowerBI <- paste0("select MEASURE_NAME, EXPRESSION, MEASUREGROUP_NAME ",
  "from `$$SYSTEM.MDSHEMA_MEASURES")
getQueryPowerBIData <- f_query_datamodel(queryPowerBI, connection_db)

```

```

str(getQueryPowerBIData)

# Example 3
# Escape double quotes so that it can run via PowerShell
queryPowerBI <- paste0("evaluate(summarizecolumns('TopMovies'[Rank], ",
  "'TopMovies'[Title],\\\"\\\"\"Value\\\"\\\"\", ",
  "TopMovies[Avg Metascore]))")
getQueryPowerBIData <- f_query_datamodel(queryPowerBI, connection_db)
str(getQueryPowerBIData)

# Example 4
# Return results from multiple EVALUATE.
# Remember to put white spaces after statements like DEFINE and EVALUATE
# the code runs
queryPowerBI <- paste0(
  "DEFINE ",
  "VAR test_average = CALCULATE(AVERAGE('TopMovies'[imdbRating])) ",
  "VAR test_median = CALCULATE(MEDIAN('TopMovies'[imdbRating])) ",
  "EVALUATE ",
  "  ROW( ",
  "    \\\"\\\"\"MinRuntime\\\"\\\"\", CALCULATE(MIN('TopMovies'[Runtime])),",
  "    \\\"\\\"\"MaxRuntime\\\"\\\"\", CALCULATE(MAX('TopMovies'[Runtime])),",
  "    \\\"\\\"\"average\\\"\\\"\", test_average) ",
  "EVALUATE ",
  "  ROW(",
  "    \\\"\\\"\"MinRuntime\\\"\\\"\", CALCULATE(MIN('TopMovies'[Runtime])),",
  "    \\\"\\\"\"MaxRuntime\\\"\\\"\", CALCULATE(MAX('TopMovies'[Runtime])),",
  "    \\\"\\\"\"median\\\"\\\"\", test_median)"
)
getQueryPowerBIData <- f_query_datamodel(queryPowerBI, connection_db)
str(getQueryPowerBIData[[1]])
str(getQueryPowerBIData[[2]])

# Example 5
# Use single quotes when white space occurs in table name
# Note that single quotation marks don't have to be escaped for
# 'PowerShell'.
queryPowerBI <- "evaluate 'Genre Bridge'"
getQueryPowerBIData <- f_query_datamodel(queryPowerBI, connection_db)
str(getQueryPowerBIData)

# Example 6
# Statement that won't work.
queryPowerBI <- "hello, world"
getQueryPowerBIData <- f_query_datamodel(queryPowerBI, connection_db)
getQueryPowerBIData

## End(Not run)

```

Description

The byte sequence of a '.json' file within a '.pbix' is retrieved, cleaned by removing ASCII control characters and written to a temporary file. An attempt is made to read the temporary file as '.json'. If reading the temporary file as '.json' fails, a second attempt is made. For the second attempt, specific data within the '.json' file is included and a temporary file is written. The temporary file is read as '.json'.

Usage

```
f_read_any_json(input_file_pbix, input_file, gsub1, gsub2)
```

Arguments

input_file_pbix	Path of the input '.pbix'.
input_file	Path of '.json' file in collection of files.
gsub1	Text to select for replacement (i.e. text to exclude).
gsub2	Text to replace selected text (i.e. text to include).

Value

Layout as '.json'.

Author(s)

Don Diproto

See Also

Uses: [f_get_pbix_fir](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
```



```

url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
url <- paste0(url_pt1, url_pt2)
req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Run the function
gsub__1 <- paste0(".*sections")
gsub__2 <- "{\\"id\\":0,\\"sections"
test <- f_read_any_json(pathFileSample, "Report/Layout",
                        gsub__1, gsub__2)

## End(Not run)

```

f_read_layout

Read Layout as '.json' from the '.pbix' Collection of Files

Description

The byte sequence of Layout within a '.pbix' is retrieved, cleaned by removing ASCII control characters and written to a temporary file. An attempt is made to read the temporary file as '.json'. If reading the temporary file as '.json' fails, a second attempt is made. For the second attempt, specific data within the '.json' file is included and a temporary file is written. The temporary file is read as '.json'.

Usage

```
f_read_layout(input_file_pbix, gsub1, gsub2)
```

Arguments

input_file_pbix	Path of the input '.pbix'.
gsub1	Text to select for replacement (i.e. text to exclude).
gsub2	Text to replace selected text (i.e. text to include).

Value

json: Layout

Author(s)

Don Diprto

See Also

Uses: [f_read_any_json](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

gsub__1 <- paste0(".*sections")
gsub__2 <- "{\\\"id\\\":0,\\\"sections\"
# Run the function
test <- f_read_layout(pathFileSample, gsub__1, gsub__2)

## End(Not run)
```

f_remove_file

Remove a File in the '.pbix' Collection of Files

Description

A '.pbix' is decompressed, making its collection of files available for manipulation. A file is removed from the collection. Files remaining in the collection are (1) compressed to form a modified '.pbix' and (2) deleted.

Usage

```
f_remove_file(input_file_pbix, collection_files_pbix, output_pbix, file_remove)
```

Arguments

input_file_pbix Path of the input '.pbix'.

collection_files_pbix Directory of the collection of files.

output_pbix Path of the modified '.pbix'.

file_remove Name of file in the collection of files to be removed.

Value

None

Author(s)

Don Diproto

See Also

Uses: [f_decompress_pbix](#), [f_compress_pbix](#).

Examples

```
## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(),"functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
  pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

pathFileSampleMod <- file.path(temp_dir, "sample_modified_f20.pbix")
dirFileSampleMod <- file.path(temp_dir, "sample_modified_f20")
```

```

if(file.exists(pathFileSampleMod)) {
  file.remove(pathFileSampleMod)
}
if(dir.exists(dirFileSampleMod)) {
  unlink(dirFileSampleMod, recursive = TRUE)
}
# Run the function
f_remove_file(pathFileSample, dirFileSampleMod, pathFileSampleMod,
"DataModel")

## End(Not run)

```

f_search_xml

Search '.xml' Within 'DataMashup'

Description

A helper for searches of an '.xml' within 'DataMashup'.

Usage

```
f_search_xml(input_xml, search_string, option)
```

Arguments

input_xml '.xml' that will be searched
search_string String for search
option option for the type of '.xml' search.

Value

Search results

Author(s)

Don Diproto

Examples

```

## Not run:
# Get dummy data -----
# Create a temporary directory
temp_dir <- file.path(tempdir(), "functionTest")
if(!dir.exists(temp_dir)) {
  dir.create(temp_dir)
}
sample_file_name <- "OR_sample_func.pbix"
pathFileSample <- file.path(temp_dir, sample_file_name)

```

```
# See if dummy data already exists in temporary directory
parent_temp_dir <- dirname(temp_dir)
existing_file <- list.files(parent_temp_dir,
pattern = sample_file_name, recursive = TRUE, full.names = TRUE)

# Download the sample .pbix if it doesn't exist
if (length(existing_file) == 0) {
  url_pt1 <- "https://github.com/KoenVerbeeck/PowerBI-Course/blob/"
  url_pt2 <- "master/pbix/TopMovies.pbix?raw=true"
  url <- paste0(url_pt1, url_pt2)
  req <- download.file(url, destfile = pathFileSample, mode = "wb")
} else {
  pathFileSample <- existing_file[1]
}
# Do stuff -----

# Get the start and end positions
test <- f_get_dama_xml_details(pathFileSample)
xml_start <- (test[[1]][1]/2) + 1
xml_end <- test[[3]][1]
# Get the .xml Within DataMashup
output <- f_get_dama_xml(pathFileSample, xml_start, xml_end)
# Pattern for query names
get_line <- "//ItemLocation[ItemType = \"Formula\"]//ItemPath"
# Run the function
f_search_xml(output, get_line, 1)

## End(Not run)
```

Index

f_clean_under_the_hood, 2
f_compress_pbix, 4, 27
f_decompress_pbix, 5, 27
f_extract_images, 6
f_get_connections, 8
f_get_dama, 9, 11, 12
f_get_dama_file, 10, 13, 15, 17
f_get_dama_index, 12, 13, 15, 17
f_get_dama_m, 13
f_get_dama_xml, 14
f_get_dama_xml_data, 16
f_get_dama_xml_details, 17
f_get_pbix_fir, 7, 9, 18, 24
f_get_pbix_info, 7, 20
f_query_datamodel, 21
f_read_any_json, 23, 26
f_read_layout, 25
f_remove_file, 3, 26
f_search_xml, 16, 28