Package: verifyr2 (via r-universe)

November 3, 2026

Type Package

Title Compare and Verify File Contents

Description Extendable 'R6' file comparison classes, including a 'shiny' app for combining the comparison functionality into a file comparison application. The package idea originates from pharma companies' drug development processes, where statisticians and statistical programmers need to review and compare different versions of the same outputs and datasets. The package implementation itself is not tied to any specific industry and can be used in any context for easy file comparisons between different file version sets.

Version 1.3.0

License MIT + file LICENSE

Depends R (>= 4.1.0)

Imports base64enc, diffobj, dplyr, htmltools, jsonlite, magrittr, mime, R6, rappdirs, shiny, stringr, striprtf, tibble

Suggests DT, fs, magick, pkgload, pdftools, purrr, readxl, rmarkdown, shinyjs, shinyFiles, shinytest2, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.3

Collate 'FileComparator.R' 'BinaryFileComparator.R' 'Config.R' 'Debugger.R' 'TxtFileComparator.R' 'FileComparatorFactory.R' 'ImgFileComparator.R' 'PdfFileComparator.R' 'TxtWithImagesComparator.R' 'RtfFileComparator.R' 'XlsxFileComparator.R' 'list_files.R' 'list_folder_files.R' 'run_example.R' 'utils-pipe.R'

Config/pak/sysreqs cmake make libicu-dev libuv1-dev zlib1g-dev

Repository https://ejuhjav.r-universe.dev

Date/Publication 2026-08-05 06:08:14 UTC

RemoteUrl https://github.com/ejuhjav/verifyr2

RemoteRef HEAD

RemoteSha df3bb7dc5638ea8fde8abf394aa598bf7c30c706

Contents

BinaryFileComparator . . . 2
Config . . . 4
create_comparator . . . 7
Debugger . . . 7
FileComparator . . . 9
ImgFileComparator . . . 12
list_files . . . 15
list_folder_files . . . 16
PdfFileComparator . . . 16
RtfFileComparator . . . 18
run_example . . . 20
TxtFileComparator . . . 20
TxtWithImagesFileComparator . . . 22
XlsxFileComparator . . . 24

Index **27**

`BinaryFileComparator` *BinaryFileComparator.R*

Description

BinaryFileComparator.R

BinaryFileComparator.R

Details

Fallback comparator for binary files without any specific definied comparator.

Super class

`verifyr2::FileComparator` -> `BinaryFileComparator`

Methods

Public methods:

- `BinaryFileComparator$vrf_contents()`
- `BinaryFileComparator$vrf_contents_inner()`
- `BinaryFileComparator$vrf_summary_inner()`
- `BinaryFileComparator$vrf_details_inner()`
- `BinaryFileComparator$clone()`

Method `vrf_contents()`**:** Method for getting the single file contents for the comparison. This method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents with the rows matching the omit string excluded. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn not be called directly by the user.

Usage:

```
BinaryFileComparator$vrf_contents(file, config, omit)
```

Arguments:

`file` file for which to get the contents

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_contents_inner()`**:** Method for getting the inner part for the file contents query. The method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents with the rows matching the omit string excluded. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn not be called directly by the user.

Usage:

```
BinaryFileComparator$vrf_contents_inner(contents, config, omit)
```

Arguments:

`contents` file contents

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_summary_inner()`**:** Method for comparing the inner part for the details query. The method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents processed for empty spaces and omit terms if applicable. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn not be called directly by the user.

Usage:

```
BinaryFileComparator$vrf_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_inner()`**:** Method for comparing the inner part for the details query. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
BinaryFileComparator$vrf_details_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
BinaryFileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# The normal way for creating a comparator would be to call the generic
# factory method verifyr2::create_comparator that will automatically create
# the correct comparator instance based on the file types.

file1 <- 'my_file1.bin'
file2 <- 'my_file2.bin'
comparator <- verifyr2::create_comparator(file1, file2)

# If needed, an explicit comparator can be created as well.

file1 <- 'my_file1.bin'
file2 <- 'my_file2.bin'
comparator <- BinaryFileComparator$new(file1, file2)
```

`Config` *Config.R*

Description

Config.R

Config.R

Details

Class for manging the library configuration options. Creates the default configuration without any source file, populates partial or missing config elements, stores the config file to local machine, and provides easy access methods for setting and getting config values.

Public fields

`schema` configuration schema

`config` current configuration data

`path` configuration json file path

Methods

Public methods:

- Config$new()
- Config$get_schema_item()
- Config$get()
- Config$set()
- Config$save()
- Config$get_default_config()
- Config$get_default_schema()
- Config$clone()

Method `new()`**:** Constructor for initializing the configuration. Checks the local machine for existing configuration file is load_config = TRUE. Ensures that all the project configuration values are included.

Usage:

```
Config$new(load_config = TRUE, config_path = NULL)
```

Arguments:

`load_config` load configuration from local machine if available

`config_path` location of the used/stored configuration json file

Method `get_schema_item()`**:** Method for getting schema item based on configuration key. Configuration item children are separated with a dot in the key notation.

Usage:

```
Config$get_schema_item(key)
```

Arguments:

`key` configuration property key for which to get the schema item

Method `get()`**:** Method for getting configuration value based on configuration key. Configuration item children are separated with a dot in the key notation.

Usage:

```
Config$get(key)
```

Arguments:

`key` configuration property key for which to get the value

Method `set()`**:** Method for setting configuration value based on configuration key. Configuration item children are separated with a dot in the key notation.

Usage:

```
Config$set(key, value)
```

Arguments:

`key` configuration property key for which to get the value

`value` value to set for the specified configuration key

Method `save()`**:** Method for saving the current configuration data into local machine.

Usage:

```
Config$save()
```

Method `get_default_config()`**:** Helper method for getting configuration default values. These default values will be used in the configuration in case the configuration properties are not present previously.

Usage:

```
Config$get_default_config()
```

Method `get_default_schema()`**:** Method for getting the full configuration schema. Apart from the configuration data, the schema contains property descriptions as well as all possible values for the configuration properties.

Usage:

```
Config$get_default_schema()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Config$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# Creates the configuration instance. Checks automatically if there is
# a previously stored configuration json file available for usage. Note
# that you don't need to explicitly define the config file path for the
# Config instance - by default the config file will be searched from and
# written in user-specific configuration directory for the package.

path   <- tempfile(fileext = ".json")
config <- Config$new(config_path = path)

# Getting and setting configuration values

value <- config$get("defailts.mode")
config$set("details.mode", "full")

# Saving the current configuration to local machine (to tmp folder with
# the given explicit file path in initialization).

config$save()
```

`create_comparator` *FileComparatorFactory.R*

Description

Factory method for creating comparator instance based on the given two files.

Usage

```
create_comparator(file1, file2)
```

Arguments

`file1`	first file to compare
`file2`	second file to compare

Examples

```
# instantiating the compared files
file1 <- "file1.rtf"
file2 <- "file2.rtf"

# instantiating the configuration
config <- Config$new()

# instantiating a new comparator instance for every comparison:
comparator <- verifyr2::create_comparator(file1, file2)

# calling the summary and details comparison methods
comparator$vrf_summary(config = config)
comparator$vrf_details(config = config)
```

`Debugger` *Debugger.R*

Description

Debugger.R

Debugger.R

Details

Class for managing the library debugging. Tracks the debugged method execution times and prints out the full debugging information only when the main debugging instance is stopped.

Public fields

`stack` local property for storing debugging labels and start times

Methods

Public methods:

- `Debugger$new()`
- `Debugger$open_debug()`
- `Debugger$add_debug()`
- `Debugger$close_debug()`
- `Debugger$print_debug_tree()`
- `Debugger$clone()`

Method `new()`**:** Constructor for initializing the Debugger instance.

Usage:

```
Debugger$new()
```

Method `open_debug()`**:** Method for opening new debug instance to the current debugging stack. Stores also the start time for execution time calculation.

Usage:

```
Debugger$open_debug(label)
```

Arguments:

`label` debugging message

Method `add_debug()`**:** Method for adding a new debug string under the currently open debug instance.

Usage:

```
Debugger$add_debug(label)
```

Arguments:

`label` debugging message

Method `close_debug()`**:** Method for closing a debug instance from the current debugging stack. If the stopped debug instance is the main level one, the whole debug data is printed out to console. If the stopped debug instance is not the main level one, calculates the execution time of current debug instance and updates the stack data.

Usage:

```
Debugger$close_debug()
```

Method `print_debug_tree()`**:** Recursive method for printing out the current debug stack items and recursively all the item children. This method is called for the whole stack once the topmost debug instance is stopped.

Usage:

```
Debugger$print_debug_tree(entry, depth = 0)
```

Arguments:

`entry` current debug level being processed for printing
`depth` current processing depth for printing indentation

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Debugger$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# Creates a debugger instance.

debugger <- Debugger$new()

# Opening and closing debugs in multileveled function calls.

function1 <- function() {
  debugger$open_debug("function1")
  function2()
  debuger$close_debug()
}

function2 <- function() {
  debugger$open_debug("function2")
  Sys.sleep(1)
  debugger$close_debug()
}

# This will produce the following printout to the console after the
# function1 finishes
#
# - 'function1' (execution time 7 ms)
#   - 'function2' (execution time 5 ms)
```

`FileComparator` *FileComparator.R*

Description

FileComparator.R

FileComparator.R

Details

Comparator 'abstract' class containing the generic comparison methods and handling for high level checks (like file existence). This class should never be instantiated - doing that and calling the comparison methods will lead to error.

Public fields

`file1` file1

`file2` file2

`file1_contents_list` file1 contents

`file2_contents_list` file2 contents

`summary_comparison` summary comparison result

`details_comparison` details comparison result

`debugger` debugger instance

Methods

Public methods:

- `FileComparator$new()`
- `FileComparator$vrf_summary()`
- `FileComparator$vrf_details()`
- `FileComparator$vrf_summary_inner()`
- `FileComparator$vrf_details_inner()`
- `FileComparator$vrf_details_supported()`
- `FileComparator$vrf_option_value()`
- `FileComparator$vrf_open_debug()`
- `FileComparator$vrf_add_debug()`
- `FileComparator$vrf_add_debug_files()`
- `FileComparator$vrf_close_debug()`
- `FileComparator$clone()`

Method `new()`**:** Initialize a FileComparator instance

Usage:

```
FileComparator$new(file1 = NULL, file2 = NULL)
```

Arguments:

`file1` First file to compare.

`file2` Second file to compare.

Method `vrf_summary()`**:** Method for comparing the file summary information. This method is intended to be implemented only this class level. For comparator specific rules, the internal method vrf_summary_inner should be customized on lower levels instead.

Usage:

```
FileComparator$vrf_summary(config, omit = NULL)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details()`**:** Method for comparing the file details information. This method is intended to be implemented only this class level. For comparator specific rules, the internal method vrf_summary_inner should be customized on lower levels instead.

Usage:

```
FileComparator$vrf_details(config, omit = NULL)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_summary_inner()`: "Abstract" method for comparing the inner part for the summary. This method has to be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
FileComparator$vrf_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_inner()`: "Abstract" method for comparing the inner part for the detailsThis method has to be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
FileComparator$vrf_details_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_supported()`: Inherited method for indicating whether detailed comparison is available with the current comparator. Returns an empty string if the comparator is is supported, otherwise a string that will be concatenated with the summary string.

Usage:

```
FileComparator$vrf_details_supported(config)
```

Arguments:

`config` configuration values

Method `vrf_option_value()`: Method for getting specific value from the config In the initial version, returns 'NA' if null con is passed.

Usage:

```
FileComparator$vrf_option_value(config, key)
```

Arguments:

`config` configuration values

`key` key to search from the parameters

Method `vrf_open_debug()`: Wrapper method for the opening a new debugging instance with Debugger class if debugging is enabled in config class. Creates the used debugger instance if needed.

Usage:

```
FileComparator$vrf_open_debug(message, config)
```

Arguments:

`message` message to debug to console

`config` configuration values

Method `vrf_add_debug()`**:** Wrapper method for the adding a new debugging message with Debugger class.

Usage:

```
FileComparator$vrf_add_debug(message)
```

Arguments:

`message` message to debug to console

Method `vrf_add_debug_files()`**:** Special method for adding the compared files into debugger stack.

Usage:

```
FileComparator$vrf_add_debug_files()
```

Method `vrf_close_debug()`**:** Wrapper method for the stopping (closing) current debugging instance with Debugger class.

Usage:

```
FileComparator$vrf_close_debug()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
FileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`ImgFileComparator` *ImgFileComparator.R*

Description

ImgFileComparator.R

ImgFileComparator.R

Details

Specialiced comparator for image file (jpg, jpef, png) comparison. This comparator contains the custom handling for handling only img content part for the comparison.

Super classes

`verifyr2::FileComparator` -> `verifyr2::BinaryFileComparator` -> `ImgFileComparator`

Public fields

`image1_raw` extracted raw data for image1.

`image2_raw` extracted raw data for image2.

Methods

Public methods:

- `ImgFileComparator$new()`
- `ImgFileComparator$vrf_details_inner()`
- `ImgFileComparator$vrf_details_inner_from_raw()`
- `ImgFileComparator$vrf_details_inner_from_files()`
- `ImgFileComparator$vrf_details_supported()`
- `ImgFileComparator$vrf_render_image_diff()`
- `ImgFileComparator$clone()`

Method `new()`**:** Initialize a ImgFileComparator instance

Usage:

```
ImgFileComparator$new(file1 = NULL, file2 = NULL, raw1 = NULL, raw2 = NULL)
```

Arguments:

`file1` First file to compare.

`file2` Second file to compare.

`raw1` First image in raw format to compare.

`raw2` Second image in raw format to compare.

Method `vrf_details_inner()`**:** Method for comparing the inner part for the details query. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
ImgFileComparator$vrf_details_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_inner_from_raw()`**:** Internal method for comparing the earlier populated raw image contents in details and generating the difference highlight image in case differences are found.

Usage:

```
ImgFileComparator$vrf_details_inner_from_raw(config)
```

Arguments:

`config` configuration values

Method `vrf_details_inner_from_files()`**:** Method for comparing the inner part for the details query with the file names as the base arguments. This is a part of a group of image processing functions that work with different image abstractions (file, image, raw image). These methods are intended to improve the performance so that best suiting method version can be used depending on what data is available from the earlier method calls to the same comparator instance.

Usage:

```
ImgFileComparator$vrf_details_inner_from_files(config)
```

Arguments:

`config` configuration values

Method `vrf_details_supported()`**:** Inherited method for indicating whether detailed comparison is available with the current comparator. Returns an empty string if the comparator is is supported, otherwise a string that will be concatenated with the summary string.

Usage:

```
ImgFileComparator$vrf_details_supported(config)
```

Arguments:

`config` configuration values

Method `vrf_render_image_diff()`**:** Method for rendering image differences as HTML.

Usage:

```
ImgFileComparator$vrf_render_image_diff(image1, image2, image3)
```

Arguments:

`image1` first image as base64 decoded string

`image2` second image as base64 decoded string

`image3` differences image as base64 decoded string

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
ImgFileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# The normal way for creating a comparator would be to call the generic
# factory method verifyr2::create_comparator that will automatically create
# the correct comparator instance based on the file types.

file1 <- 'my_file1.jpg'
file2 <- 'my_file2.jpg'
comparator <- verifyr2::create_comparator(file1, file2)

# If needed, an explicit comparator can be created as well.

file1 <- 'my_file1.png'
file2 <- 'my_file2.png'
```

```
comparator <- ImgFileComparator$new(file1, file2)

# This comparator has also second explicit creation method that is used
# by the library for processing embedded image contents.

image1_raw <- 'raw hex vector data'
image2_raw <- 'raw hex vector data'
comparator <- ImgFileComparator$new(NULL, NULL, image1_raw, image2_raw)
```

`list_files` *One row list for two distinct files*

Description

`list_files` List single file row based on the explicit parameter files. This is a conveniency function for building same structure list for direct two file comparison case.

Usage

```
list_files(file1, file2)
```

Arguments

`file1`	character, giving the the full file path of the first file
`file2`	character, giving the the full file path of the second file

Value

Returns a tibble, `selected_files` with 2 columns `file1`, `file2`

Examples

```
path1 <- "/extdata/base_files/file2_additional_rows.rtf"
file1 <- paste0(fs::path_package(path1, package = "verifyr2"))

path2 <- "/extdata/compare_files/file3_changed_rows.rtf"
file2 <- paste0(fs::path_package(path2, package = "verifyr2"))

verifyr2::list_files(file1, file2)
```

list_folder_files *List files that exist in two folders*

Description

list_folder_files List files that exist in two folders with a specific name pattern

Usage

```
list_folder_files(folder1, folder2, pattern = NULL)
```

Arguments

folder1	character, giving the the full file path and name of the folder where first files are stored (required)
folder2	character, giving the the full file path and name of the folder where second new files are stored (required)
pattern	character, limit the files to be listed to contain a specific pattern (optional)

Value

Returns a tibble, selected_files with 2 columns file1, file2

Examples

```
folder1 <- paste0(fs::path_package("/extdata/base_files/",
                                   package = "verifyr2"))

folder2 <- paste0(fs::path_package("/extdata/compare_files/",
                                   package = "verifyr2"))

verifyr2::list_folder_files(folder1, folder2, "base")
```

PdfFileComparator *PdfFileComparator.R*

Description

PdfFileComparator.R

PdfFileComparator.R

Details

Specialiced comparator for PDF file comparison. This comparator contains the custom handling for handling only PDF content part for the comparison.

Super classes

verifyr2::FileComparator -> verifyr2::BinaryFileComparator -> verifyr2::TxtFileComparator -> PdfFileComparator

Methods

Public methods:

- PdfFileComparator$vrf_contents()
- PdfFileComparator$vrf_summary_inner()
- PdfFileComparator$vrf_details_supported()
- PdfFileComparator$clone()

Method `vrf_contents()`**:** Method for getting the single file contents for the comparison. The method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents with the rows matching the omit string excluded. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
PdfFileComparator$vrf_contents(file, config, omit)
```

Arguments:

`file` file for which to get the contents

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_summary_inner()`**:** Method for comparing the file contents on the summary level. When the detailed (pdftools-based) comparison is disabled, a text-based diff of the raw pdf bytes would give platform-dependent results, so a deterministic binary (byte-level) comparison is used instead. Otherwise the normal text-based summary comparison is used. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
PdfFileComparator$vrf_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_supported()`**:** Inherited method for indicating whether detailed comparison is available with the current comparator. Returns an empty string if the comparator is is supported, otherwise a string that will be concatenated with the summary string.

Usage:

```
PdfFileComparator$vrf_details_supported(config)
```

Arguments:

`config` configuration values

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
PdfFileComparator$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
# The normal way for creating a comparator would be to call the generic
# factory method verifyr2::create_comparator that will automatically create
# the correct comparator instance based on the file types.

file1 <- 'my_file1.pdf'
file2 <- 'my_file2.pdf'
comparator <- verifyr2::create_comparator(file1, file2)

# If needed, an explicit comparator can be created as well.

file1 <- 'my_file1.pdf'
file2 <- 'my_file2.pdf'
comparator <- PdfFileComparator$new(file1, file2)
```

`RtfFileComparator` *RtfFileComparator.R*

Description

RtfFileComparator.R

RtfFileComparator.R

Details

Specialiced comparator for RTF file comparison. This comparator contains the custom handling for handling only RTF content part for the comparison.

Super classes

`verifyr2::FileComparator` -> `verifyr2::BinaryFileComparator` -> `verifyr2::TxtFileComparator` -> `verifyr2::TxtWithImagesFileComparator` -> `RtfFileComparator`

Methods

Public methods:

- `RtfFileComparator$vrf_contents()`
- `RtfFileComparator$vrf_images()`
- `RtfFileComparator$safe_read_rtf()`

- RtfFileComparator$clone()

Method `vrf_contents()`**:** Method for getting the single file contents for the comparison. The method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents with the rows matching the omit string excluded. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

For RtfComparator, only the RTF file content part is returned for comparison.

Usage:

```
RtfFileComparator$vrf_contents(file, config, omit)
```

Arguments:

`file` file for which to get the contents

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_images()`**:** "Abstract" method for getting the raw image hex vector array from the given source file.

Usage:

```
RtfFileComparator$vrf_images(file, config)
```

Arguments:

`file` file for which to get the embedded image details

`config` configuration values

Method `safe_read_rtf()`**:** RTF content reading method with some fallback options to fix simple file content issues.

Usage:

```
RtfFileComparator$safe_read_rtf(file)
```

Arguments:

`file` file for which to get the embedded image details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
RtfFileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# The normal way for creating a comparator would be to call the generic
# factory method verifyr2::create_comparator that will automatically create
# the correct comparator instance based on the file types.

file1 <- 'my_file1.rtf'
file2 <- 'my_file2.rtf'
```

```
comparator <- verifyr2::create_comparator(file1, file2)

# If needed, an explicit comparator can be created as well.

file1 <- 'my_file1.rtf'
file2 <- 'my_file2.rft'
comparator <- RtfFileComparator$new(file1, file2)
```

`run_example` *Call for shiny example where the user can test verifyr2 package functions*

Description

`verifyr2::run_example` returns simple Shiny App where user can see how the verifyr2 functions work

Usage

```
run_example(debug = FALSE)
```

Arguments

debug	option to override debug configuration (TRUE only)

`TxtFileComparator` *TextFileComparator.R*

Description

TextFileComparator.R

TextFileComparator.R

Details

Fallback comparator for text files without any specific definied comparator. This comparator contains the methods for doing basic comparisons on raw text contents.

Super classes

`verifyr2::FileComparator` -> `verifyr2::BinaryFileComparator` -> `TxtFileComparator`

Methods

Public methods:

- TxtFileComparator$vrf_summary_inner()
- TxtFileComparator$vrf_details_inner()
- TxtFileComparator$vrf_contents_inner()
- TxtFileComparator$vrf_details_supported()
- TxtFileComparator$vrf_binary_summary_inner()
- TxtFileComparator$clone()

Method `vrf_summary_inner()`**:** Method for comparing the inner part for the details query. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
TxtFileComparator$vrf_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_inner()`**:** Method for comparing the inner part for the details query. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
TxtFileComparator$vrf_details_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_contents_inner()`**:** Method for getting the inner part for the file contents query. The method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents processed for empty spaces and omit terms if applicable. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
TxtFileComparator$vrf_contents_inner(contents, config, omit)
```

Arguments:

`contents` file contents

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_supported()`**:** Inherited method for indicating whether detailed comparison is available with the current comparator. Returns an empty string if the comparator is is supported, otherwise a string that will be concatenated with the summary string.

Usage:

```
TxtFileComparator$vrf_details_supported(config)
```

Arguments:

`config` configuration values

Method `vrf_binary_summary_inner()`**:** Helper for performing a binary (byte-level) summary comparison. This delegates to the BinaryFileComparator summary logic and is used by specialised comparators as a deterministic, platform-independent fallback when their detailed (text-based) comparison is disabled. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
TxtFileComparator$vrf_binary_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
TxtFileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# The normal way for creating a comparator would be to call the generic
# factory method verifyr2::create_comparator that will automatically create
# the correct comparator instance based on the file types.

file1 <- 'my_file1.txt'
file2 <- 'my_file2.txt'
comparator <- verifyr2::create_comparator(file1, file2)

# If needed, an explicit comparator can be created as well.

file1 <- 'my_file1.lst'
file2 <- 'my_file2.lst'
comparator <- TxtFileComparator$new(file1, file2)
```

`TxtWithImagesFileComparator` *TxtWithImageFileComparator.R*

Description

TxtWithImageFileComparator.R

TxtWithImageFileComparator.R

Details

"Abstract" comparator for txt based comparator classes that can additionally contain embedded images. This abstraction level contains generic logic for handling embedded images and storing the related data.

Super classes

verifyr2::FileComparator -> verifyr2::BinaryFileComparator -> verifyr2::TxtFileComparator -> TxtWithImagesFileComparator

Public fields

`file1_images_raw` local property for storing image1 raw data

`file2_images_raw` local property for storing image2 raw data

Methods

Public methods:

- TxtWithImagesFileComparator$new()
- TxtWithImagesFileComparator$vrf_summary_inner()
- TxtWithImagesFileComparator$vrf_details_inner()
- TxtWithImagesFileComparator$vrf_images()
- TxtWithImagesFileComparator$hex2raw()
- TxtWithImagesFileComparator$clone()

Method `new()`**:** Initialize a TxtWithImagesFileComparator instance

Usage:

```
TxtWithImagesFileComparator$new(file1 = NULL, file2 = NULL)
```

Arguments:

`file1` First file to compare.

`file2` Second file to compare.

Method `vrf_summary_inner()`**:** Method for comparing the inner part for the details query. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
TxtWithImagesFileComparator$vrf_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_inner()`**:** Method for comparing the inner part for the details query. This method can be overwritten by more specialized comparator classes. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
TxtWithImagesFileComparator$vrf_details_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_images()`**:** "Abstract" method for getting the raw image hex vector array from the given source file.

Usage:

```
TxtWithImagesFileComparator$vrf_images(file)
```

Arguments:

`file` file for which to get the embedded image details

Method `hex2raw()`**:** Internal helper method for converting a hex string to raw vector.

Usage:

```
TxtWithImagesFileComparator$hex2raw(hex_string)
```

Arguments:

`hex_string` hexadecimal string to be converted to raw vector

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
TxtWithImagesFileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`XlsxFileComparator` *XlsxFileComparator.R*

Description

XlsxFileComparator.R

XlsxFileComparator.R

Details

Specialised comparator for Excel file comparison. This comparator contains the custom handling for reading Excel file contents for comparison.

Super classes

`verifyr2::FileComparator` -> `verifyr2::BinaryFileComparator` -> `verifyr2::TxtFileComparator` -> `XlsxFileComparator`

Methods

Public methods:

- `XlsxFileComparator$vrf_contents()`
- `XlsxFileComparator$vrf_summary_inner()`
- `XlsxFileComparator$vrf_details_supported()`
- `XlsxFileComparator$clone()`

Method `vrf_contents()`**:** Method for getting the single file contents for the comparison. The method returns the file contents in two separate vectors inside a list. The first vector is the file contents and the second one is the file contents with the rows matching the omit string excluded. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

For XlsxFileComparator, each sheet is read and every row is flattened to a single line. Each line is annotated with the sheet name and, when a header row is present, each cell is written as `Column=Value` so that any detected difference clearly indicates the sheet, the row and the column it originates from. Whether the first row is treated as a header is controlled by the `xlsx.header` configuration option: `"yes"` (default) treats it as a header, `"no"` treats every row as data and uses positional column names instead, flagging the sheet marker accordingly.

Usage:

```
XlsxFileComparator$vrf_contents(file, config, omit)
```

Arguments:

`file` file for which to get the contents

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_summary_inner()`**:** Method for comparing the file contents on the summary level. When the detailed (readxl-based) comparison is disabled, a text-based diff of the raw xlsx bytes would give platform-dependent results, so a deterministic binary (byte-level) comparison is used instead. Otherwise the normal text-based summary comparison is used. This method is intended to be called only by the comparator classes in the processing and shouldn't be called directly by the user.

Usage:

```
XlsxFileComparator$vrf_summary_inner(config, omit)
```

Arguments:

`config` configuration values

`omit` string pattern to omit from the comparison

Method `vrf_details_supported()`**:** Inherited method for indicating whether detailed comparison is available with the current comparator. Returns an empty string if the comparator is supported, otherwise a string that will be concatenated with the summary string.

Usage:

```
XlsxFileComparator$vrf_details_supported(config)
```

Arguments:

`config` configuration values

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
XlsxFileComparator$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# The normal way for creating a comparator would be to call the generic
# factory method verifyr2::create_comparator that will automatically create
# the correct comparator instance based on the file types.

file1 <- 'my_file1.xlsx'
file2 <- 'my_file2.xlsx'
comparator <- verifyr2::create_comparator(file1, file2)

# If needed, an explicit comparator can be created as well.

file1 <- 'my_file1.xlsx'
file2 <- 'my_file2.xlsx'
comparator <- XlsxFileComparator$new(file1, file2)
```

Index

BinaryFileComparator, 2

Config, 4
create_comparator, 7

Debugger, 7

FileComparator, 9

ImgFileComparator, 12

list_files, 15
list_folder_files, 16

PdfFileComparator, 16

RtfFileComparator, 18
run_example, 20

TxtFileComparator, 20
TxtWithImagesFileComparator, 22

verifyr2::BinaryFileComparator, *12*, *17*, *18*, *20*, *23*, *24*
verifyr2::FileComparator, *2*, *12*, *17*, *18*, *20*, *23*, *24*
verifyr2::TxtFileComparator, *17*, *18*, *23*, *24*
verifyr2::TxtWithImagesFileComparator, *18*

XlsxFileComparator, 24