Package 'gDRimport'

November 13, 2026

Type Package

Title Package for handling the import of dose-response data

Version 1.11.15

Date 2026-10-29

Description The package is a part of the gDR suite. It helps to prepare raw drug response data for downstream processing. It mainly contains helper functions for importing/loading/validating dose-response data provided in different file formats.

License Artistic-2.0

LazyLoad yes

Depends R (>= 4.2)

Imports assertthat, BumpyMatrix, checkmate, data.table, futile.logger, gDRutils (>= 1.7.1), magrittr, methods, MultiAssayExperiment, readxl, rio, S4Vectors, stats, stringi, SummarizedExperiment, tibble, tools, utils, XML, yaml, openxlsx, qs2

Suggests BiocStyle, CoreGx, gDRtestData (>= 1.7.1), gDRstyle (>= 1.7.1), knitr, PharmacoGx, purrr, testthat

URL https://github.com/gdrplatform/gDRimport, https://gdrplatform.github.io/gDRimport/

BugReports https://github.com/gdrplatform/gDRimport/issues

biocViews Software, Infrastructure, DataImport

VignetteBuilder knitr

ByteCompile TRUE

Roxygen list(markdown = TRUE)

SwitchrLibrary gDRimport

DeploySubPath gDRimport

Encoding UTF-8

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.1

git_url https://git.bioconductor.org/packages/gDRimport

git_branch devel

git_last_commit 52e7a93

git_last_commit_date 2026-11-02

Repository Bioconductor 3.24

Date/Publication 2026-11-12

Author Arkadiusz Gladki [aut, cre] (ORCID:
<https://orcid.org/0000-0002-7059-6378>),
Bartosz Czech [aut] (ORCID: <https://orcid.org/0000-0002-9908-3007>),
Marc Hafner [aut] (ORCID: <https://orcid.org/0000-0003-1337-7598>),
Sergiu Mocanu [aut],
Dariusz Scigocki [aut],
Allison Vuong [aut],
Luca Gerosa [aut] (ORCID: <https://orcid.org/0000-0001-6805-9410>),
Janina Smola [aut] (ORCID: <https://orcid.org/0009-0007-4347-7748>)

Maintainer Arkadiusz Gladki <gladki.arkadiusz@gmail.com>

Contents

gDRimport-package 4
.assert_suggested_packages 5
.check_against_single_template_sheet 5
.check_file_structure 6
.createPseudoData 6
.extractDoseResponse 7
.extract_or_create_assay 7
.fill_empty_wells 8
.get_plate_size 8
.removeNegatives 9
.standardize_untreated_values 9
are_template_sheets_valid 9
check_metadata_against_spaces 10
check_metadata_field_names 10
check_metadata_headers 11
check_metadata_names 11
check_metadata_req_col_names 12
convert_LEVEL5_prism_to_gDR_input 12
convert_LEVEL6_prism_to_gDR_input 13
convert_MAE_to_PSet 14
convert_pset_to_df 14
correct_template_sheets 15
detect_file_format 16
enhance_raw_edited_EnVision_df 16
fix_typos_with_reference 17
gdr_test_data-class 17
getPSet 18
get_df_from_raw_edited_EnVision_df 19
get_df_from_raw_unedited_EnVision_df 19
get_EnVision_properties 20
get_excel_sheet_names 20
get_exception_data 21
get_expected_template_sheets 21
get_import_provenance 22

get_plate_info_from_template_xlsx . . . 22
get_test_D300_data . . . 23
get_test_data . . . 23
get_test_EnVision_data . . . 24
get_test_Tecan_data . . . 24
get_test_tsv_data . . . 25
get_xl_sheets . . . 25
import_D300 . . . 26
is_readable_v . . . 27
load_data . . . 27
load_long_table . . . 28
load_manifest . . . 29
load_results . . . 29
load_results_EnVision . . . 30
load_results_EnVision_new . . . 30
load_results_Incucyte . . . 31
load_results_Tecan . . . 31
load_results_tsv . . . 32
load_templates . . . 32
load_templates_tsv . . . 33
load_templates_xlsx . . . 33
manifest_path . . . 34
mgrepl . . . 34
parse_D300_xml . . . 35
read_EnVision_delim . . . 35
read_EnVision_xlsx . . . 36
read_excel_to_dt . . . 36
read_in_EnVision_file . . . 37
read_in_manifest_file . . . 37
read_in_results_Tecan . . . 38
read_in_result_files . . . 38
read_in_template_sheet_xlsx . . . 39
read_in_template_xlsx . . . 39
read_in_tsv_template_files . . . 40
read_ref_data . . . 40
result_path . . . 41
save_drug_info_per_well . . . 41
setEnvForPSet . . . 42
standardize_record_values . . . 42
template_path . . . 43
validate_gdr_input . . . 43
validate_template_xlsx . . . 45

Index **46**

gDRimport-package *gDRimport: Package for handling the import of dose-response data*

Description

The package is a part of the gDR suite. It helps to prepare raw drug response data for downstream processing. It mainly contains helper functions for importing/loading/validating dose-response data provided in different file formats.

Value

package help page

Note

To learn more about functions start with `help(package = "gDRimport")`

Author(s)

Maintainer: Arkadiusz Gladki <gladki.arkadiusz@gmail.com> (ORCID)

Authors:

- Arkadiusz Gladki <gladki.arkadiusz@gmail.com> (ORCID)
- Bartosz Czech (ORCID)
- Marc Hafner (ORCID)
- Sergiu Mocanu
- Dariusz Scigocki
- Allison Vuong
- Luca Gerosa (ORCID)
- Janina Smola (ORCID)

See Also

Useful links:

- https://github.com/gdrplatform/gDRimport
- https://gdrplatform.github.io/gDRimport/
- Report bugs at https://github.com/gdrplatform/gDRimport/issues

.assert_suggested_packages *Assert that suggested packages required by a function are installed*

Description

The PharmacoSet conversion functions are the only part of gDRimport that needs CoreGx and PharmacoGx, and those two carry 37 further dependencies, so they are suggested rather than imported. Call this before touching their API.

Usage

```
.assert_suggested_packages(pkgs, purpose)
```

Arguments

pkgs	character vector of package names.
purpose	a short description of the functionality, used in the error message.

Value

NULL invisibly.

.check_against_single_template_sheet *Evaluate if template file with single sheet is present, if the name of the sheet is correct and if it can be fixed*

Description

get sheets for given set of XLS files

Usage

```
.check_against_single_template_sheet(ts)
```

Arguments

ts	list with template sheets info

Value

logical flag

.check_file_structure *Check the structure of raw data*

Description

Check the structure of raw data

Usage

```
.check_file_structure(
  df,
  filename,
  sheet_name,
  readout_offset,
  n_row,
  n_col,
  bcode_idx,
  bcode_col
)
```

Value

NULL invisibly.

.createPseudoData *Add in pseudo-data for duration and cell reference division time*

Description

Add in pseudo-data for duration and cell reference division time

Usage

```
.createPseudoData(dt)
```

Value

data.table

.extractDoseResponse *Get dose and viability readouts and melt into large data table*

Description

Get dose and viability readouts and melt into large data table

Usage

```
.extractDoseResponse(pset)
```

Value

data.table with dose-response data

.extract_or_create_assay *Extracts an assay from a SummarizedExperiment object or creates a new one if it does not exist*

Description

This function takes a SummarizedExperiment object and an assay name as input. If the specified assay already exists in the SummarizedExperiment object, it is returned. Otherwise, a new assay with the specified name is created and added to the SummarizedExperiment object. The new assay is initialized with NA values. This is useful for when multiple Summarized Experiments in a given MAE do not have the same assays. And it is necessary to have the same assays in all Summarized Experiments in order to convert the MAE to a PSet.

Usage

```
.extract_or_create_assay(SE, assay_name)
```

Arguments

SE	A SummarizedExperiment object
assay_name	A character string specifying the name of the assay to extract or create

Value

A SummarizedExperiment object with the specified assay

.fill_empty_wells *Correct plates with not fully filled readout values*

Description

Correct plates with not fully filled readout values

Usage

```
.fill_empty_wells(
  df,
  plate_rows,
  data_rows,
  exp_row,
  exp_col,
  numeric_regex = "^\\d+$"
)
```

Value

data.table with corrected plates data

.get_plate_size *Get plate size*

Description

Get plate size

Usage

```
.get_plate_size(df)
```

Details

All plate sizes assume 1.5x nrows = ncolumns.

Value

charvec with plate dims

.removeNegatives *Remove negative viabilities*

Description

Remove negative viabilities

Usage

```
.removeNegatives(dataset)
```

Value

data.table with positive values in column `ReadoutValue`

.standardize_untreated_values *Standardize untreated values to ignore cases*

Description

Standardize untreated values to ignore cases

Usage

```
.standardize_untreated_values(df)
```

Value

data.table with standardized untreated values

are_template_sheets_valid *are template sheet valid?*

Description

are template sheet valid?

Usage

```
are_template_sheets_valid(ts)
```

Arguments

`ts`	list with (per file) template sheets

Value

logical flag

See Also

get_xl_sheets

`check_metadata_against_spaces` *Check metadata against spaces*

Description

Check metadata against spaces

Usage

```
check_metadata_against_spaces(corrected_names, df_name)
```

Arguments

`corrected_names`	a charvec with corrected colnames of df
`df_name`	a name of data.table ("" by default)

Value

a charvec with corrected colnames of df

`check_metadata_field_names` *Check metadata field names*

Description

Check metadata field names

Usage

```
check_metadata_field_names(corrected_names, df_name)
```

Arguments

`corrected_names`	a charvec with corrected colnames of df
`df_name`	a name of data.table ("" by default)

Value

a charvec with corrected colnames of df

check_metadata_headers

Check whether metadata headers are correct and make fixes if needed

Description

Check whether metadata headers are correct and make fixes if needed

Usage

```
check_metadata_headers(corrected_names, df_name)
```

Arguments

corrected_names	a charvec with corrected colnames of df
df_name	a name of data.table ("" by default)

Value

a charvec with corrected colnames of df

check_metadata_names *check_metadata_names*

Description

Check whether all metadata names are correct

Usage

```
check_metadata_names(col_df, df_name = "", df_type = NULL)
```

Arguments

col_df	a character with colnames of df
df_name	a name of data.table ("" by default)
df_type	a type of a data.table (NULL by default)

Value

a charvec with corrected colnames of df

Examples

```
td <- get_test_data()
m_file <- manifest_path(td)
m_data <- read_excel_to_dt(m_file)
result <- check_metadata_names(col_df = colnames(m_data))
```

`check_metadata_req_col_names` *Check metadata for required column names*

Description

Check metadata for required column names

Usage

```
check_metadata_req_col_names(col_df, df_name, df_type)
```

Arguments

`col_df`	a charvec with corrected colnames of df
`df_name`	a name of data.table ("" by default)
`df_type`	a type of a data.table (NULL by default)

Value

`NULL` invisibly.

`convert_LEVEL5_prism_to_gDR_input` *Load, convert and process the level 5 PRISM data into a gDR input*

Description

Load, convert and process the level 5 PRISM data into a gDR input

Usage

```
convert_LEVEL5_prism_to_gDR_input(
  prism_data_path,
  meta_data_path,
  readout_min = 1.03
)
```

Arguments

`prism_data_path`	path to PRISM LEVEL5 csv file with data
`meta_data_path`	path to metadata file describing all cancer models/cell lines which are referenced by a dataset contained within the DepMap portal
`readout_min`	minimum ReadoutValue

Value

`data.table` object with input data for gDR pipeline

Examples

```
prism_data <- system.file("testdata/prism_sa.csv", package = "gDRimport")
prism_meta <- system.file("testdata/prism_model.csv", package = "gDRimport")
convert_LEVEL5_prism_to_gDR_input(prism_data, prism_meta)
```

`convert_LEVEL6_prism_to_gDR_input` *Load, convert and process the level 6 PRISM data into a gDR input*

Description

Load, convert and process the level 6 PRISM data into a gDR input

Usage

```
convert_LEVEL6_prism_to_gDR_input(
  prism_data_path,
  cell_line_data_path,
  treatment_data_path,
  meta_data_path,
  readout_min = 1.03
)
```

Arguments

`prism_data_path`	path to PRISM LEVEL6 csv file with collapsed log fold change data
`cell_line_data_path`	path to cell line info data
`treatment_data_path`	path to collapsed treatment info data
`meta_data_path`	path to metadata file describing all cancer models/cell lines which are referenced by a dataset contained within the DepMap portal
`readout_min`	minimum ReadoutValue

Value

`data.table` object with input data for gDR pipeline

Examples

```
prism_data_path <- system.file("testdata/prism_collapsed_LOGFC.csv", package = "gDRimport")
cell_line_data_path <- system.file("testdata/prism_cell_lines.csv", package = "gDRimport")
treatment_data_path <- system.file("testdata/prism_treatment.csv", package = "gDRimport")
prism_meta <- system.file("testdata/prism_model.csv", package = "gDRimport")
convert_LEVEL6_prism_to_gDR_input(prism_data_path, cell_line_data_path, treatment_data_path, prism_meta)
```

convert_MAE_to_PSet *Convert MultiAssayExperiment to TreatmentResponseExperiment*

Description

This function converts a MultiAssayExperiment generated by gDR into a TreatmentResponseExperiment for use in the PharmacoGx package. The resulting PharmacoSet can be used for pharmacogenomic analysis of drug response.

Usage

```
convert_MAE_to_PSet(mae, pset_name)
```

Arguments

mae	A MultiAssayExperiment object generated by gDR.
pset_name	A character string specifying the name of the resulting PharmacoSet object.

Value

A PharmacoSet object.

Examples

```
# Convert a MultiAssayExperiment object to a PharmacoSet object
m <- 20
n <- 10
rnames <- LETTERS[1:m]
cnames <- letters[1:n]
ref_gr_value <-  matrix(runif(m * n), nrow = m, ncol = n, dimnames = list(rnames, cnames))
se <- SummarizedExperiment::SummarizedExperiment(assays = list(RefGRvalue = ref_gr_value),
                                                 rowData = S4Vectors::DataFrame(rnames),
                                                 colData = S4Vectors::DataFrame(cnames))
mae <- MultiAssayExperiment::MultiAssayExperiment(experiments = list("single-agent" = se))
convert_MAE_to_PSet(mae, "my_pset")
```

convert_pset_to_df *Convert a PharmacoSet to a data.table that is prepare for input into gDR pipeline*

Description

Convert a PharmacoSet to a data.table that is prepare for input into gDR pipeline

Usage

```
convert_pset_to_df(pharmacoset, run_parallel = TRUE, workers = 2L)
```

Arguments

pharmacoset	PharmacoSet object
run_parallel	logical, TRUE (default) if to run functions in Parallel, FALSE to run in serial
workers	integer, number of workers defaults to 2L if run_parallel is TRUE

Value

data.table of PharmacoSet's dose response data with column names aligned with gDR standard

Author(s)

Jermiah Joseph – collaboration with BHKLab

Examples

```
pset <- suppressMessages(getPSet(
  "Tavor_2020",
  psetDir = system.file("extdata/pset", package = "gDRimport"),
  use_local_PSets_list = TRUE
))
dt <- convert_pset_to_df(pset)
gDRutils::reset_env_identifiers()
```

---

correct_template_sheets

Correct names of the template sheets (if required)

Description

Correct names of the template sheets (if required)

Usage

```
correct_template_sheets(tfiles)
```

Arguments

tfiles	charvec with paths to template files

Value

charvec with paths to corrected sheet names

detect_file_format *Detect format of results data*

Description

Detect format of results data

Usage

```
detect_file_format(results_file)
```

Arguments

results_file	path to results data

Value

string of the detected file format

Examples

```
td2 <- get_test_Tecan_data()
detect_file_format(td2$r_files[1])
```

enhance_raw_edited_EnVision_df *Enhance raw edited EnVision data.table*

Description

Enhance raw edited EnVision data.table

Usage

```
enhance_raw_edited_EnVision_df(df, barcode_col, headers)
```

Arguments

df	raw data.table
barcode_col	column number for barcode data
headers	list with the headersa

Value

data.table derived from EnVision data

fix_typos_with_reference *Fix typos using reference data*

Description

Fix typos using reference data Evaluate given list of ids and try to update them

Usage

```
fix_typos_with_reference(
  data,
  ref,
  method = c("exact", "grepl", "adist"),
  fix_underscores = FALSE
)
```

Arguments

data	list of charvec(s) or charvec with data
ref	charvec with reference data
method	charvec type of the method to be used 'exact' is used to find identical entries from 'ref' in the data (after corrections and uppercase'ing) 'grepl' is used to find entries from 'ref' that might be somehow pre- or post- fixed
fix_underscores	logical flag fix the issues with underscores in data identfiers?

Value

list or charvec with corrected data

gdr_test_data-class *gDR Test Data object*

Description

Object class gdr_test_data is build by function get_test_data()

Value

object class gdr_test_data with primary test data

Slots

`manifest_path` character, path to manifest file

`result_path` character, path(s) to results file

`template_path` character, path(s) to data.table with template data

`ref_m_df` character, data.table with manifest data

`ref_r1_r2` character, path to reference file with raw data for treated & untreated

`ref_r1` character, path to reference file with raw data for treated

`ref_t1_t2` character, path to reference template file with treated & untreated data

`ref_t1` character, path to reference template file with treated data

---

`getPSet` *Get PharmacoSet*

Description

Get PharmacoSet

Usage

```
getPSet(
  pset_name,
  psetDir = getwd(),
  canonical = FALSE,
  timeout = 600,
  use_local_PSets_list = FALSE
)
```

Arguments

`pset_name`	string with the name of the PharmacoSet
`psetDir`	string with the temporary directory for the PharmacoSet
`canonical`	logical flag indicating if the PSet canonical
`timeout`	maximum number of seconds allowed for PSet download
`use_local_PSets_list`	logical flag if PSets list should be used from local. If FALSE PSets list will be taken from web.

Value

PharmacoSet object

Examples

```
suppressMessages(getPSet(
  "Tavor_2020",
  psetDir = system.file("extdata/pset", package = "gDRimport"),
  use_local_PSets_list = TRUE
))
```

get_df_from_raw_edited_EnVision_df

Get final results (as a data.table) from raw edited EnVision data.table

Description

Get final results (as a data.table) from raw edited EnVision data.table

Usage

```
get_df_from_raw_edited_EnVision_df(
  df,
  barcode_idx,
  barcode_col,
  n_row,
  n_col,
  fname,
  sheet_name,
  headers
)
```

Arguments

df	raw data.table
barcode_idx	numeric vector with barcode indices
barcode_col	column number for barcode data
n_row	number of rows
n_col	number of columns
fname	file name
sheet_name	name of the Excel sheet
headers	list with the headersa

Value

data.table derived from EnVision data

get_df_from_raw_unedited_EnVision_df

Get final results (as a data.table) from raw unedited EnVision data.table

Description

Get final results (as a data.table) from raw unedited EnVision data.table

Usage

```
get_df_from_raw_unedited_EnVision_df(df, n_row, n_col, barcode_col)
```

Arguments

df	raw data.table
n_row	number of rows
n_col	number of columns
barcode_col	column number for barcode data

Value

data.table derived from EnVision data

---

get_EnVision_properties *Get properties of EnVision data*

Description

This function return properties of EnVision data

Usage

```
get_EnVision_properties(results.list, fname)
```

Arguments

results.list	list with EnVision data
fname	name of the input file

Value

list with EnVision propertiesa

---

get_excel_sheet_names *get Excel sheets names for a charvec of files for non-Excel files return 0*

Description

get Excel sheets names for a charvec of files for non-Excel files return 0

Usage

```
get_excel_sheet_names(fls)
```

Arguments

fls	charvec with file pathsa

Value

list with one element per file with sheet names or 0 (for non-Excel file)

get_exception_data *get exception data*

Description

get exception data

Usage

```
get_exception_data(status_code = NULL)
```

Arguments

status_code	A numeric value

Value

A data.table row with exception data or all exceptions

Examples

```
get_exception_data(1)
get_exception_data()
```

get_expected_template_sheets *Get names of the sheets expected in templates xlsx*

Description

Get names of the sheets expected in templates xlsx

Usage

```
get_expected_template_sheets(type = c("all", "core", "optional"))
```

Arguments

type	charvec type of the sheets

Value

string with type of the sheets

get_import_provenance *Where gDR input data came from*

Description

load_data and the PRISM converters attach a record of their inputs to the table they return: the gDRimport version, the reader (instrument), every input file with its role, size and md5, and the corrections gDRimport applied on its own. gDRcore keeps the record in the provenance of the processed object (provenance$steps$import).

Usage

```
get_import_provenance(x)
```

Arguments

x	the data element returned by load_data, or a table returned by a PRISM converter.

Value

The record (a list), or NULL when x carries none.

Examples

```
td <- get_test_data()
l_data <- load_data(manifest_path(td), template_path(td), result_path(td))
get_import_provenance(l_data$data)$files
```

get_plate_info_from_template_xlsx *Get plate info from template xlsx*

Description

Get plate info from template xlsx

Usage

```
get_plate_info_from_template_xlsx(template_file, Gnumber_idx, idx)
```

Arguments

template_file	character, file path(s) to template(s)
Gnumber_idx	index with Gnumber data
idx	template file index

Value

list with plate info

get_test_D300_data *get test D300 data*

Description

get test D300 data

Usage

```
get_test_D300_data()
```

Value

list with with input data (manifest/template/result paths) and related reference data (qs2 file paths)

Examples

```
get_test_D300_data()
```

get_test_data *get primary test data*

Description

get primary test data

Usage

```
get_test_data()
```

Value

object class "gdr_test_data" with with input data (manifest/template/result paths) and related reference data (qs2 file paths)

Examples

```
get_test_data()
```

get_test_EnVision_data *get test EnVision data*

Description

get test EnVision data

Usage

```
get_test_EnVision_data()
```

Value

list with with input data (manifest/template/result paths) and related reference data (.qs2 file paths)

Examples

```
get_test_EnVision_data()
```

get_test_Tecan_data *get test Tecan data*

Description

get test Tecan data

Usage

```
get_test_Tecan_data()
```

Value

list with with input data (manifest/template/result paths) and related reference data (qs2 file paths)

Examples

```
get_test_Tecan_data()
```

get_test_tsv_data *get test tsv data*

Description

get test tsv data

Usage

```
get_test_tsv_data()
```

Value

list with with input data (manifest/template/result paths) and related reference data (.qs2 file paths)

Examples

```
get_test_tsv_data()
```

get_xl_sheets *Get Excel sheets*

Description

get sheets for given set of XLS files

Usage

```
get_xl_sheets(files)
```

Arguments

files	charvec with file paths

Value

named list where names are the excel filenames and the values are the sheets within each file

`import_D300` *Import D300*

Description

This functions takes a D300 file and generates corresponding template files

Usage

```
import_D300(D300_file, destination_path, metadata_file = NULL, day0 = FALSE)
```

Arguments

`D300_file`	character, file path to D300 file
`destination_path`	character, path to folder where template files will be generated
`metadata_file`	character, file path to file with mapping from D300 names to Gnumbers. Defaults to NULL.
`day0`	logical, if TRUE, creates a template file for Day 0 data filled with vehicles in addition to the standard plates. Defaults to FALSE.

Details

For example, wells treated with 2 drugs in combination will result in 4 sheets per plate.

- Sheet 1: Drug 1
- Sheet 2: Conc of Drug 1
- Sheet 3: Drug 2
- Sheet 4: Conc of Drug 2

Value

Create one Excel file per plate. Each sheet in each plate file describes the drugs and corrresponding concentrations of what was tested in each well.

Examples

```
td3 <- get_test_D300_data()[["f_96w"]]
o_path <- file.path(tempdir(), "td3")
dir.create(o_path)
import_D300(td3$d300, o_path, td3$Gnum)
list.files(o_path)
unlink(o_path, recursive = TRUE)
```

is_readable_v *is_readable_v Check if all paths in vector are readable*

Description

is_readable_v Check if all paths in vector are readable

Usage

```
is_readable_v(paths)
```

Arguments

paths	a character with path(s)

Value

NULL invisibly.

Examples

```
td2 <- get_test_Tecan_data()
is_readable_v(td2$r_files)
```

load_data *Load data*

Description

This functions loads and checks the data file(s)

Usage

```
load_data(
  manifest_file,
  df_template_files,
  results_file,
  instrument = "EnVision"
)
```

Arguments

manifest_file	character, file path(s) to manifest(s)
df_template_files	data.table, with datapaths and names of results file(s) or character with file path of templates file(s)
results_file	data.table, with datapaths and names of results file(s) or character with file path of results file(s)
instrument	character

Value

a list with three data.tables for manifest/treatment and results

Examples

```
td <- get_test_data()
l_tbl <- load_data(manifest_path(td), template_path(td), result_path(td))
```

`load_long_table` *Load long table*

Description

Loads a long/tidy `data.table` from a delimited text file (CSV/TSV) and checks that it contains the columns required by the gDR pipeline. This offers an input path that does not rely on GeneDataScreeneR: the data can come from a custom export or be provided directly by the user in tabular form, matching the shape of the `data_imported` object used downstream.

Usage

```
load_long_table(long_table_file)
```

Arguments

`long_table_file`
: character, path to a single CSV/TSV file with the long table. The field separator is detected automatically.

Details

Required columns follow the current gDR identifiers: `Gnumber`, `clid`, `Duration` and `Concentration`. The readout column is the fixed header `ReadoutValue`: there is no `"readout"` identifier in `gDRutils`, so this name is not customisable via `get_env_identifiers()`, matching how `ReadoutValue` is used across the rest of gDRimport. Combination data may add `Gnumber_2` and `Concentration_2`. `Duration`, `Concentration` and `ReadoutValue` must be numeric.

Value

a `data.table` with the validated long table

Examples

```
path <- tempfile(fileext = ".csv")
dt <- data.table::data.table(Gnumber = "G1", clid = "CL1", Duration = 72,
                             Concentration = 1, ReadoutValue = 1000)
data.table::fwrite(dt, path)
load_long_table(path)
```

load_manifest *Load manifest*

Description

This functions loads and checks the manifest file(s)

Usage

```
load_manifest(manifest_file)
```

Arguments

manifest_file	character, file path(s) to manifest(s)

Value

list with manifest data.table and headers

Examples

```
td <- get_test_data()
ml <- load_manifest(manifest_path(td))
```

load_results *Load results*

Description

This functions loads and checks the results file(s)

Usage

```
load_results(
  df_results_files,
  instrument = "EnVision",
  headers = gDRutils::get_env_identifiers()
)
```

Arguments

df_results_files	data.table, with datapaths and names of results file(s) or character with file path of results file(s)
instrument	character; .tsv files with "EnVision" are read as "long_tsv", whether it is passed or left as the default
headers	list of headers identified in the manifest file

Value

data.table with results' data

Examples

```
td <- get_test_data()
r_df <- load_results(result_path(td))
```

---

load_results_EnVision *Load EnVision results from xlsx*

Description

This functions loads and checks the results file(s)

Usage

```
load_results_EnVision(results_file, headers = gDRutils::get_env_identifiers())
```

Arguments

results_file	character vector containing file path(s) to results file(s)
headers	list of headers identified in the manifest

Value

data.table with results data

---

load_results_EnVision_new *Load results from EnVision_new (CSV and XLSX)*

Description

This functions loads and checks the results file(s) from a new Envision instrument in the CSV or XLSX format. Supports multiple plates in a single file or multiple sheets in an Excel file by robustly checking the file structure.

Usage

```
load_results_EnVision_new(
  results_file,
  headers = gDRutils::get_env_identifiers()
)
```

Arguments

results_file	character, file path(s) to result file(s)
headers	list of headers identified in the manifest

Value

data.table with results data

`load_results_Incucyte` *Load incucyte results from plain text*

Description

This functions loads incucyte time-course cell count file

Usage

```
load_results_Incucyte(results_file, headers = gDRutils::get_env_identifiers())
```

Arguments

`results_file`	list of strings: file paths to result paths from individual plates
`headers`	list of headers identified in the manifest

Value

data.table derived from Incucyte data

`load_results_Tecan` *Load tecan results from xlsx*

Description

This functions loads and checks the results file

Usage

```
load_results_Tecan(results_file, headers = gDRutils::get_env_identifiers())
```

Arguments

`results_file`	string, file path to a result file
`headers`	list of headers identified in the manifest

Value

data.table derived from Tecan data

load_results_tsv *Load results from tsv*

Description

This functions loads and checks the results file(s)

Usage

```
load_results_tsv(results_file, headers)
```

Arguments

results_file	character, file path(s) to template(s)
headers	list of headers identified in the manifest

Value

data.table with results data

load_templates *Load templates*

Description

This functions loads and checks the template file(s)

Usage

```
load_templates(df_template_files)
```

Arguments

df_template_files	data.table, with datapaths and names of results file(s) or character with file path of templates file(s)

Value

data.table with templates data

Examples

```
td <- get_test_data()
t_df <- load_templates(template_path(td))
```

load_templates_tsv *Load templates from tsv*

Description

This functions loads and checks the template file(s)

Usage

```
load_templates_tsv(template_file, template_filename = NULL)
```

Arguments

template_file	character, file path(s) to template(s)
template_filename	character, file name(s)

Value

data.table with template data

load_templates_xlsx *Load templates from xlsx*

Description

This functions loads and checks the template file(s)

Usage

```
load_templates_xlsx(template_file, template_filename = NULL)
```

Arguments

template_file	character, file path(s) to template(s)
template_filename	character, file name(s)

Value

data.table with templates data

manifest_path *Method manifest_path*

Description

Method for object gdr_test_data - access to slot `manifest_path`

Usage

```
manifest_path(x)

## S4 method for signature 'gdr_test_data'
manifest_path(x)
```

Arguments

`x`	object class gdr_test_data

Value

value of slot `manifest_path`

Examples

```
td <- get_test_data()
manifest_file_path <- manifest_path(td)
```

mgrepl *grep wrapper to support multiple patterns*

Description

grep wrapper to support multiple patterns

Usage

```
mgrepl(patterns, x, do_unlist = TRUE, ...)
```

Arguments

`patterns`	charvec with patterns to be checked
`x`	charvec with data
`do_unlist`	logical_flag unlist the final results?
`...`	additional argument

Value

list of charvec with grep output

parse_D300_xml *Parse D300*

Description

This function parses a D300 *.tdd file (XML format) into a data.table

Usage

```
parse_D300_xml(D300_file)
```

Arguments

D300_file	string, file path to D300 .tdd file

Value

data.table representing input D300_file.

Examples

```
td3 <- get_test_D300_data()
fs <- td3[["f_96w"]]
dose_df <- parse_D300_xml(fs[["d300"]])
```

read_EnVision_delim *Read EnVision delimited text files*

Description

This function reads file from the EnVision Workstation

Usage

```
read_EnVision_delim(file, nrows = 10000, seps = c(",", "\t"))
```

Arguments

file	string to path of input file from EnVision scanner
nrows	maximum number of file rows to be processed
seps	potential field separators of the input file

Value

a list containing the data table, n_col, n_row, and if is edited

read_EnVision_xlsx *Read in single xlsx data from EnVision*

Description

Read in single xlsx data from EnVision

Usage

```
read_EnVision_xlsx(results_file, results_sheet)
```

Arguments

results_file	character, file path(s) to results file(s)
results_sheet	results sheet names

Value

data.table with results data

read_excel_to_dt *Read excel file and transorm it into data.table object*

Description

Read excel file and transorm it into data.table object

Usage

```
read_excel_to_dt(path, ...)
```

Arguments

path	path to excel file
...	other arguments that should be passed into readxl::read_excel

Value

data.table object with read excel file

Examples

```
datasets <- readxl::readxl_example("datasets.xlsx")
read_excel_to_dt(datasets)
```

read_in_EnVision_file *Read EnVision file*

Description

This function reads file from the EnVision Workstation

Usage

```
read_in_EnVision_file(file, nrows, seps)
```

Arguments

file	input file from EnVision
nrows	maximum number of file rows to be processed
seps	potential field separators of the input file

Value

list with one element per EnVisoin input file

read_in_manifest_file *read manifest files*

Description

read manifest files

Usage

```
read_in_manifest_file(manifest_file, available_formats)
```

Arguments

manifest_file	character, file path(s) to manifest(s)
available_formats	charvec with available file formats

Value

a data.table with manifest data

read_in_results_Tecan *read in Tecan data*

Description

read in Tecan data

Usage

```
read_in_results_Tecan(results_file, results_sheets, headers)
```

Arguments

results_file	string, file path to a result file
results_sheets	template sheet names
headers	list of headers identified in the manifest

Value

data.table derived from Tecan data

read_in_result_files *Read in results files*

Description

Read in results files

Usage

```
read_in_result_files(results_file, results_filename, headers)
```

Arguments

results_file	data.table, with datapaths and names of results file(s) or character with file path of results file(s)
results_filename	charvect with file names
headers	list of headers identified in the result files

Value

data.table with results data

read_in_template_sheet_xlsx *Read in data from xlsx template sheet*

Description

Read in data from xlsx template sheet

Usage

```
read_in_template_sheet_xlsx(template_file, template_sheets, idx, plate_info)
```

Arguments

template_file	character, file path(s) to template(s)
template_sheets	template sheet names
idx	template file index
plate_info	list with plate info

Value

data.table with template data

read_in_template_xlsx *Read in xlsx template files*

Description

Read in xlsx template files

Usage

```
read_in_template_xlsx(template_file, template_filename, template_sheets)
```

Arguments

template_file	character, file path(s) to template(s)
template_filename	character, file name(s)
template_sheets	template sheet names

Value

data.table with templates data

read_in_tsv_template_files *read in tsv template files*

Description

read in tsv template files

Usage

```
read_in_tsv_template_files(template_file, template_filename, templates)
```

Arguments

template_file	character, file path(s) to template(s)
template_filename	character, file name(s)
templates	list with templates data

Value

data.table with templates data

read_ref_data *read_ref_data*

Description

Read reference data

Usage

```
read_ref_data(inDir, prefix = "ref")
```

Arguments

inDir	a directory path of reference data
prefix	a prefix of reference file names ('ref' by default)

Value

a list of reference data

result_path *Method result_path*

Description

Method for object gdr_test_data - access to slot result_path

Usage

```
result_path(x)

## S4 method for signature 'gdr_test_data'
result_path(x)
```

Arguments

x	object class gdr_test_data

Value

value of slot result_path

Examples

```
td <- get_test_data()
result_file_path <- result_path(td)
```

save_drug_info_per_well *for each drug create a Gnumber and Concentration information for each well*

Description

for each drug create a Gnumber and Concentration information for each well

Usage

```
save_drug_info_per_well(trt_info, trt_gnumber_conc, wb, idfs)
```

Arguments

trt_info	list with treatment info
trt_gnumber_conc	list with treatment data
wb	pointer to xlsx workbook
idfs	charvec with identifiers

Value

NULL invisibly.

setEnvForPSet *Adjust environment variables to meet gDR standards*

Description

Adjust environment variables to meet gDR standards

Usage

```
setEnvForPSet()
```

Value

NULL

Examples

```
setEnvForPSet()
gDRutils::reset_env_identifiers()
```

standardize_record_values *standardize_record_values*

Description

map values to a dictionary

Usage

```
standardize_record_values(x, dictionary = DICTIONARY)
```

Arguments

x	a named array
dictionary	a named array

Value

a named array with updated names

Examples

```
standardize_record_values(c("Vehicle", "vehcle"))
```

template_path *Method template_path*

Description

Method for object gdr_test_data - access to slot `template_path`

Usage

```
template_path(x)

## S4 method for signature 'gdr_test_data'
template_path(x)
```

Arguments

x	object class gdr_test_data

Value

value of slot `template_path`

Examples

```
td <- get_test_data()
template_file_path <- template_path(td)
```

validate_gdr_input *Validate gDR input data before import*

Description

Checks a set of input files and reports every problem it finds, from unreadable files to plate-level anomalies in the readouts. Bench input (a manifest, treatment templates and raw plate-reader files) is loaded with `load_data`, so a failure reported for `load_data` is the error the import itself would raise. The checks also flag problems the loaders let through without an error: values in a Concentration sheet that are not numbers (those wells are dropped), raw plates without a barcode, duplicated wells, plates missing from the manifest or from the data, treated wells whose cell line and Duration have no control wells, drugs that appear only at concentration 0, and Duration labels that do not fit the readouts of the control wells. PRISM input, processed MultiAssayExperiments and D300 files are recognised and checked too.

Usage

```
validate_gdr_input(
  paths,
  type = c("auto", "bench", "prism", "mae", "pre_d300"),
  instrument = NULL,
  id_patterns = list(drug = "^G[0-9]{8}", cellline = NULL),
  fix = FALSE,
  output_dir = NULL
)

## S3 method for class 'gdr_validation'
print(x, ...)
```

Arguments

paths	character, paths to input files and/or directories. For a directory, the files directly inside it are used, not those in its subdirectories.
type	character, the kind of input: "auto" (detected from the files), "bench" (manifest, templates and raw plate-reader files), "prism", "mae" (a processed gDR MultiAssayExperiment saved as .qs2) or "pre_d300" (D300 .tdd files that still need import_D300).
instrument	character, the plate-reader format passed to load_data. When NULL, it is detected from the first raw file with detect_file_format.
id_patterns	list of regular expressions for identifier formats. drug is the format of a drug identifier once its batch suffix is removed (the default is the one gDRutils::remove_drug_batch uses), and cellline the format of a cell line identifier (no default). An element set to NULL switches its check off; elements left out keep their default.
fix	logical, whether to repair copies of the files in output_dir.
output_dir	character, a new or empty directory for the repaired copies; required with fix = TRUE.
x	a gdr_validation object.
...	not used.

Details

The input files are only read, never modified. With fix = TRUE they are copied to output_dir, the mechanical problems (fix tier "A") are repaired in the copies, and the copies are validated again: files re-encoded as UTF-8, spaces trimmed from file names, long tables saved as .tsv, Gnumber_1 and Concentration_1 sheets renamed, n/a and -99 cleared from templates, and manifest references that only differ from a template file by spaces or the extension rewritten. In xlsx files only the repaired cells change.

Value

An object of class gdr_validation, a list with

verdict "PASS", "WARN" or "FAIL", the most severe finding ("INFO" counts as "PASS").

data_type the kind of input that was validated.

files a data.table with one row per input file: file, path, role and format (the plate-reader format of raw files).

findings a `data.table` with one row per finding: `check`, `severity` (`"PASS"`, `"INFO"`, `"WARN"` or `"FAIL"`), `detail`, `fix_tier` (`"A"` a mechanical fix, `"B"` a fix to confirm with whoever made the files, `"C"` a decision about the experiment, `NA` when there is nothing to fix) and `status_code`, the matching row of `get_exception_data` (`NA` when there is none).

fixes character, one note per repair (empty without `fix`).

output_dir the directory of the repaired copies, `NA` without `fix`. With `fix = TRUE`, `files` and `findings` describe the copies after the repairs.

Examples

```
td <- get_test_data()
res <- validate_gdr_input(c(manifest_path(td), template_path(td), result_path(td)))
res$verdict
res$findings

# repair copies of the files instead of only reporting
fixed <- validate_gdr_input(c(manifest_path(td), template_path(td), result_path(td)),
                            fix = TRUE, output_dir = tempfile())
fixed$fixes
```

`validate_template_xlsx` *Validate template xlsx data*

Description

Validate template xlsx data

Usage

```
validate_template_xlsx(template_file, template_filename, template_sheets, idx)
```

Arguments

`template_file`	character, file path(s) to template(s)
`template_filename`	character, file name(s)
`template_sheets`	template sheet names
`idx`	template file index

Value

`NULL` invisibly.

Index

∗ **D300**
import_D300, 26
parse_D300_xml, 35
save_drug_info_per_well, 41
∗ **classes**
gdr_test_data-class, 17
∗ **correction_exception**
.check_against_single_template_sheet, 5
are_template_sheets_valid, 9
check_metadata_against_spaces, 10
check_metadata_field_names, 10
check_metadata_headers, 11
check_metadata_names, 11
check_metadata_req_col_names, 12
correct_template_sheets, 15
fix_typos_with_reference, 17
get_exception_data, 21
get_expected_template_sheets, 21
get_xl_sheets, 25
mgrepl, 34
validate_gdr_input, 43
∗ **internal**
.assert_suggested_packages, 5
.check_against_single_template_sheet, 5
.check_file_structure, 6
.createPseudoData, 6
.extractDoseResponse, 7
.extract_or_create_assay, 7
.fill_empty_wells, 8
.get_plate_size, 8
.removeNegatives, 9
.standardize_untreated_values, 9
are_template_sheets_valid, 9
check_metadata_against_spaces, 10
check_metadata_field_names, 10
check_metadata_headers, 11
check_metadata_req_col_names, 12
correct_template_sheets, 15
enhance_raw_edited_EnVision_df, 16
gDRimport-package, 4
get_EnVision_properties, 20
get_excel_sheet_names, 20
get_plate_info_from_template_xlsx, 22
get_xl_sheets, 25
load_results_EnVision_new, 30
manifest_path, 34
mgrepl, 34
read_EnVision_delim, 35
read_EnVision_xlsx, 36
read_in_EnVision_file, 37
read_in_manifest_file, 37
read_in_result_files, 38
read_in_results_Tecan, 38
read_in_template_sheet_xlsx, 39
read_in_template_xlsx, 39
read_in_tsv_template_files, 40
read_ref_data, 40
result_path, 41
template_path, 43
validate_template_xlsx, 45
∗ **load_files**
enhance_raw_edited_EnVision_df, 16
get_df_from_raw_edited_EnVision_df, 19
get_df_from_raw_unedited_EnVision_df, 19
get_EnVision_properties, 20
get_excel_sheet_names, 20
get_import_provenance, 22
get_plate_info_from_template_xlsx, 22
load_data, 27
load_long_table, 28
load_manifest, 29
load_results, 29
load_results_EnVision, 30
load_results_EnVision_new, 30
load_results_Incucyte, 31
load_results_Tecan, 31
load_results_tsv, 32
load_templates, 32
load_templates_tsv, 33
load_templates_xlsx, 33

read_EnVision_delim, 35
read_EnVision_xlsx, 36
read_in_EnVision_file, 37
read_in_manifest_file, 37
read_in_result_files, 38
read_in_results_Tecan, 38
read_in_template_sheet_xlsx, 39
read_in_template_xlsx, 39
read_in_tsv_template_files, 40
validate_template_xlsx, 45
∗ **methods**
manifest_path, 34
result_path, 41
template_path, 43
∗ **prism_conversion**
convert_LEVEL5_prism_to_gDR_input, 12
convert_LEVEL6_prism_to_gDR_input, 13
∗ **pset_conversion**
convert_MAE_to_PSet, 14
convert_pset_to_df, 14
getPSet, 18
setEnvForPSet, 42
∗ **test_data_class**
gdr_test_data-class, 17
get_test_data, 23
manifest_path, 34
result_path, 41
template_path, 43
∗ **test_data**
get_test_D300_data, 23
get_test_EnVision_data, 24
get_test_Tecan_data, 24
get_test_tsv_data, 25
∗ **utils**
detect_file_format, 16
is_readable_v, 27
read_excel_to_dt, 36
read_ref_data, 40
standardize_record_values, 42
.assert_suggested_packages, 5
.check_against_single_template_sheet, 5
.check_file_structure, 6
.createPseudoData, 6
.extractDoseResponse, 7
.extract_or_create_assay, 7
.fill_empty_wells, 8
.get_plate_size, 8
.removeNegatives, 9
.standardize_untreated_values, 9

are_template_sheets_valid, 9

check_metadata_against_spaces, 10
check_metadata_field_names, 10
check_metadata_headers, 11
check_metadata_names, 11
check_metadata_req_col_names, 12
convert_LEVEL5_prism_to_gDR_input, 12
convert_LEVEL6_prism_to_gDR_input, 13
convert_MAE_to_PSet, 14
convert_pset_to_df, 14
correct_template_sheets, 15

detect_file_format, 16, *44*

enhance_raw_edited_EnVision_df, 16

fix_typos_with_reference, 17

gdr_test_data-class, 17
gDRimport *(*gDRimport-package*)*, 4
gDRimport-package, 4
get_df_from_raw_edited_EnVision_df, 19
get_df_from_raw_unedited_EnVision_df, 19
get_EnVision_properties, 20
get_excel_sheet_names, 20
get_exception_data, 21, *45*
get_expected_template_sheets, 21
get_import_provenance, 22
get_plate_info_from_template_xlsx, 22
get_test_D300_data, 23
get_test_data, 23
get_test_data(), *17*
get_test_EnVision_data, 24
get_test_Tecan_data, 24
get_test_tsv_data, 25
get_xl_sheets, 25
getPSet, 18

import_D300, 26, *44*
is_readable_v, 27

load_data, *22*, 27, *43, 44*
load_long_table, 28
load_manifest, 29
load_results, 29
load_results_EnVision, 30
load_results_EnVision_new, 30
load_results_Incucyte, 31
load_results_Tecan, 31
load_results_tsv, 32
load_templates, 32
load_templates_tsv, 33

load_templates_xlsx, 33

manifest_path, 34
manifest_path,gdr_test_data-method
(manifest_path), 34
mgrepl, 34

parse_D300_xml, 35
print.gdr_validation
(validate_gdr_input), 43

read_EnVision_delim, 35
read_EnVision_xlsx, 36
read_excel_to_dt, 36
read_in_EnVision_file, 37
read_in_manifest_file, 37
read_in_result_files, 38
read_in_results_Tecan, 38
read_in_template_sheet_xlsx, 39
read_in_template_xlsx, 39
read_in_tsv_template_files, 40
read_ref_data, 40
result_path, 41
result_path,gdr_test_data-method
(result_path), 41

save_drug_info_per_well, 41
setEnvForPSet, 42
standardize_record_values, 42

template_path, 43
template_path,gdr_test_data-method
(template_path), 43

validate_gdr_input, 43
validate_template_xlsx, 45