Package ‘mgi.report.reader’

May 8, 2026

Title Read Mouse Genome Informatics Reports

Version 0.1.3

Description Provides readers for easy and consistent importing of Mouse Genome Informatics (MGI) report files: <https://www.informatics.jax.org/downloads/reports/index.html>. These data are provided by Baldarelli RM, Smith CL, Ringwald M, Richardson JE, Bult CJ, Mouse Genome Informatics Group (2024) <doi:10.1093/genetics/iyae031>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

Depends R (>= 2.10)

LazyData true

Imports data.table, dplyr, httr2, memoise, rlang, stringr, tibble, vroom

URL https://www.pattern.institute/mgi.report.reader/, https://github.com/patterninstitute/mgi.report.reader/

BugReports https://github.com/patterninstitute/mgi.report.reader/issues

Config/Needs/website rmarkdown, patterninstitute/chic

Suggests testthat (>= 3.0.0), tidyr

Config/testthat/edition 3

NeedsCompilation no

Author Ramiro Magno [aut, cre] (ORCID: <https://orcid.org/0000-0001-5226-3441>),
David Shaw [aut] (ORCID: <https://orcid.org/0000-0002-9922-9934>),
Isabel Duarte [aut] (ORCID: <https://orcid.org/0000-0003-0060-2936>),
Ismail Gbadamosi [aut] (ORCID: <https://orcid.org/0000-0002-1932-7519>),
Ali Jawaid [aut] (ORCID: <https://orcid.org/0000-0002-5126-6744>),
Nencki Institute of Experimental Biology [fnd],
University of Algarve [fnd],
The Jackson Laboratory [fnd],
Pattern Institute [cph, fnd]

Maintainer Ramiro Magno `<rmagno@pattern.institute>`

Repository CRAN

Date/Publication 2024-07-07 14:50:02 UTC

Contents

`chromosomes` *Mouse chromosomes*

Description

`chromosomes()` returns mouse chromosome names.

Usage

```
chromosomes(autosomal = TRUE, sexual = TRUE, mitochondrial = TRUE)
```

Arguments

`autosomal`	Whether to include the autosomal chromosomes (1 thru 19).
`sexual`	Whether to include the sexual chromosomes (X and Y).
`mitochondrial`	Whether to include the mitochondrial chromosome (MT).

Value

A character vector of mouse chromosome names, or a subset thereof, or an empty character vector.

Examples

```
# All chromosomes.
chromosomes()

# Autosomal chromosomes.
chromosomes(autosomal = TRUE, sexual = FALSE, mitochondrial = FALSE)
```

`feature_types` *Genome Feature types*

Description

`feature_types()` returns different types of gene and genome features. For feature type definitions, see `?feature_type_definitions`.

Usage

```
feature_types()
```

Value

A character vector of feature types' names.

Examples

```
feature_types()
```

`feature_type_definitions` *Genome Feature Type Definitions*

Description

A dataset containing different types of gene and genome features along with their Sequence Ontology (SO) identifiers and definitions.

Usage

```
feature_type_definitions
```

Format

A tibble with 71 rows and 3 variables:

feature_type Character. The type of gene or genome feature.

so_id Character. The Sequence Ontology identifier associated with the feature type.

definition Character. The definition of the feature type.

Source

The table in `https://www.informatics.jax.org/userhelp/GENE_feature_types_help.shtml` and a few other terms found in MGI reports.

Examples

```
print(feature_type_definitions, n = Inf)
```

`marker_types` *Genetic marker types*

Description

`marker_types()` returns MGI marker types. See marker_type_definitions for the meaning of each type.

Usage

```
marker_types()
```

Value

A character vector.

Examples

```
marker_types()
```

`marker_type_definitions` *Genetic Marker Type Definitions*

Description

A dataset of marker types definitions.

Use instead `marker_types()` for the marker type names as a single character vector.

Usage

```
marker_type_definitions
```

Format

A tibble with 10 rows and 2 variables:

`marker_type`	Character. The type of genetic marker.
`definition`	Character. The definition of the marker type.

Source

The cross-references in the entry definition for marker at MGI glossary: `https://www.informatics.jax.org/glossary/marker/`.

Examples

```
print(marker_type_definitions, n = Inf)
```

`open_marker_id_in_mgi` *Browse MGI markers identifiers online*

Description

`open_marker_id_in_mgi()` launches the web browser and opens a tab for each MGI accession identifier on the Mouse Genome Informatics web interface: `https://www.informatics.jax.org`.

Usage

```
open_marker_id_in_mgi(marker_id)
```

Arguments

`marker_id`	A character vector. MGI accession identifiers.

Value

Returns `TRUE` if successful, or `FALSE` otherwise. But note that this function is run for its side effect of launching the browser.

Examples

```
# Read about Acta1 (actin alpha 1, skeletal muscle) online.
open_marker_id_in_mgi("MGI:87902")

# `open_marker_id_in_mgi()` is vectorized, so you can open multiple pages.
# NB: think twice if you really need to open many tabs at once.
open_marker_id_in_mgi(c("MGI:87902", "MGI:87909"))
```

`open_marker_symbol_in_mgi` *Browse MGI markers symbols online*

Description

`open_marker_symbol_in_mgi()` launches the web browser and opens a tab for each MGI symbol on the Mouse Genome Informatics web interface: `https://www.informatics.jax.org`.

Usage

```
open_marker_symbol_in_mgi(marker_symbol)
```

Arguments

`marker_symbol`	A character vector. MGI marker symbols.

Value

Returns `TRUE` if successful, or `FALSE` otherwise. But note that this function is run for its side effect of launching the browser.

Examples

```
# Read about Acta1 (actin alpha 1, skeletal muscle) online.
open_marker_symbol_in_mgi("Acta1")

# `open_marker_symbol_in_mgi()` is vectorized, so you can open multiple pages.
# NB: think twice if you really need to open many tabs at once.
open_marker_symbol_in_mgi(c("Acta1", "Hes1"))
```

`read_report` *Read an MGI report*

Description

`read_report()` imports data from an MGI report into R as a tidy data set.

You may call this function in two alternative ways:

- Using `report_key`: this is the easiest approach. A report key maps to a report currently hosted at MGI, e.g. `read_report("marker_list2")` reads `MRK_List2.rpt` directly from MGI server into R. See Supported Reports below for options.
- Using `report_file` and `report_type`: this approach is more flexible as you can read directly from a file or URL.

Supported Reports:

The set of currently supported reports:

```
reports
#> # A tibble: 13 x 4
#>    report_key                 report_file              report_type   report_name
#>     <chr>                        <chr>                      <chr>         <chr>
#>  1 marker_list1               MRK_List1.rpt            MRK_List1     Mouse Gene~
#>  2 marker_list2               MRK_List2.rpt            MRK_List2     Mouse Gene~
#>  3 marker_coordinates         MGI_MRK_Coord.rpt        MGI_MRK_Coord MGI Marker~
#>  4 gene_model_coordinates     MGI_Gene_Model_Coord.rpt MGI_Gene_Mod~ MGI Gene M~
#>  5 sequence_coordinates       MGI_GTGUP.gff            MGI_GTGUP     MGI Sequen~
#>  6 genbank_refseq_ensembl_ids MRK_Sequence.rpt         MRK_Sequence  MGI Marker~
#>  7 swiss_trembl_ids           MRK_SwissProt_TrEMBL.rpt MRK_SwissPro~ MGI Marker~
#>  8 swiss_prot_ids             MRK_SwissProt.rpt        MRK_SwissProt MGI Marker~
#>  9 gene_trap_ids              MRK_GeneTrap.rpt         MRK_GeneTrap  MGI Marker~
#> 10 ensembl_ids                MRK_ENSEMBL.rpt          MRK_ENSEMBL   MGI Marker~
#> 11 biotype_conflicts          MGI_BioTypeConflict.rpt  MGI_BioTypeC~ MGI Marker~
#> 12 primers                    PRB_PrimerSeq.rpt        PRB_PrimerSeq MGI Marker~
#> 13 interpro_domains           MGI_InterProDomains.rpt  MGI_InterPro~ InterPro d~
```

Usage

```
read_report(
  report_key = NULL,
  report_file = NULL,
  report_type = NULL,
  n_max = Inf
)
```

Arguments

`report_key`	A character vector. A key used to uniquely refer to an MGI report.
`report_file`	A character vector. The file path or URL to an MGI report file.
`report_type`	A character vector. The type of an MGI report.
`n_max`	Maximum number of lines to read.

Value

A tibble with report data in tidy format. The set of variables is dependent on the specific report requested:

- For `"marker_list1"`, see `vignette("marker_list1")`.
- For `"marker_list2"`, see `vignette("marker_list2")`.
- For `"marker_coordinates"`, see `vignette("marker_coordinates")`.
- For `"gene_model_coordinates"`, see `vignette("gene_model_coordinates")`.
- For `"sequence_coordinates"`, see `vignette("sequence_coordinates")`.
- For `"genbank_refseq_ensembl_ids"`, see `vignette("genbank_refseq_ensembl_ids")`.
- For `"swiss_trembl_ids"`, see `vignette("swiss_trembl_ids")`.
- For `"swiss_prot_ids"`, see `vignette("swiss_prot_ids")`.
- For `"gene_trap_ids"`, see `vignette("gene_trap_ids")`.
- For `"ensembl_ids"`, see `vignette("ensembl_ids")`.
- For `"biotype_conflicts"`, see `vignette("biotype_conflicts")`.
- For `"primers"`, see `vignette("primers")`.
- For `"interpro_domains"`, see `vignette("interpro_domains")`.

`report-attributes` *Get MGI report specs by report key*

Description

Set of functions to retrieve metadata details of a MGI report.

Usage

```
report_file(report_key)

report_name(report_key)

report_type(report_key)

report_url(report_key)
```

Arguments

`report_key`	A character vector. A key used to uniquely refer to an MGI report.

Value

A character vector:

- `report_file()`: report file name as hosted in `https://www.informatics.jax.org/downloads/reports/`.
- `report_name()`: report title.
- `report_type()`: report type.
- `report_url()`: report remote location.

Examples

```
report_file("marker_list1")

report_name("marker_list1")

report_type("marker_list1")

report_url("marker_list1")
```

`reports` *Supported MGI reports*

Description

`reports` is a data set of supported MGI reports, meaning reports that `{mgi.report.reader}` can currently read into R. To browse all reports made available by MGI visit `https://www.informatics.jax.org/downloads/reports/`.

Usage

```
reports
```

Format

A tibble of 4 variables:

`report_key` A string key used to uniquely refer to an MGI report, which is only meaningful within the context of the `{mgi.report.reader}`.

`report_file` MGI report file name as hosted at `https://www.informatics.jax.org/downloads/reports/`.

`report_type` MGI report type. The type is used internally to find the appropriate reader for parsing, and is only meaningful within the context of `{mgi.report.reader}`.

`report_name` MGI report name. Report names are taken from `https://www.informatics.jax.org/downloads/reports/index.html`.

Examples

```
reports
```

`report_example` *Report example*

Description

`report_example()` returns the local path of an example report file. These files are typically very small and are useful for demonstrations. These are mostly used in the Examples section of functions and in unit tests.

Usage

```
report_example(report_file)
```

Arguments

`report_file`	File basename.

Examples

```
report_example("MRK_List1-EX01.rpt")

report_example("MRK_List1-EX02.rpt")

report_example("MRK_List1-EX03.rpt")
```

`report_last_modified` *Report last modification date*

Description

`report_last_modified()` returns the last modified date and time of the report source: local file or remote file. If a local file, the modification date will be that indicated by the file system; if a remote file, the date of last update is that provided by HTTP header `"last-modified"`.

MGI updates its reports weekly, every Thursday. However, not all reports are updated each week. The return value of this function is the closest you will get to a versioning of MGI report files.

Usage

```
report_last_modified(tbl)
```

Arguments

`tbl`	Report data as a tibble.

Value

A last modified date-time as a POSIXct object.

Examples

```
if (FALSE) {
  markers <- read_report("marker_list1", n_max = 10L)

  # When was the report file last updated?
  report_last_modified(markers)
}
```

`report_source` *Report source*

Description

`report_source()` returns the source used to obtain the report data: a file path or an URL.

Usage

```
report_source(tbl)
```

Arguments

`tbl`	Report data as a tibble.

Value

A single string with an absolute path to a file on disk or an URL.

Examples

```
if (FALSE) {
  markers <- read_report("marker_list1", n_max = 10L)

  # Where did the data come from?
  report_source(markers)
}
```

`symbol_to_identifier` *Convert marker symbols to updated marker identifiers*

Description

`symbol_to_identifier()` remaps old marker symbols to, in-use, most up to date marker identifiers.

Usage

```
symbol_to_identifier(x, report_file = NULL, n_max = Inf)
```

Arguments

`x`	A character vector of marker symbols to be remapped.
`report_file`	The path to a MRK_List1.rpt file. Leave this as `NULL` and the function will automatically download the report from `https://www.informatics.jax.org/downloads/reports/MRK_List1.rpt`.
`n_max`	Maximum number of lines to read from the `report_file`.

Examples

```
rpt_ex01 <- report_example("MRK_List1-EX01.rpt")
read_report(report_file = rpt_ex01, report_type = "MRK_List1") |>
  dplyr::select("marker_status", "marker_symbol", "marker_id_now")

# NB:
#   - "1700024N20Rik" has two conflicting mappings, so maps to `NA`.
#   - "Hes1" is not present in MRK_List1-EX01.rpt, so maps to `NA`.
#   - "Plpbp" (official) and "Prosc" (withdrawn) both map to "MGI:1891207"

marker_symbols <- c("2200002F22Rik", "Plpbp", "Prosc", "1700024N20Rik", "Hes1")
symbol_to_identifier(x = marker_symbols, report_file = rpt_ex01)
```

`symbol_to_symbol` *Update marker symbols*

Description

`symbol_to_symbol()` remaps old marker symbols to, in-use, most up to date symbols.

Usage

```
symbol_to_symbol(x, report_file = NULL, n_max = Inf)
```

Arguments

`x`	A character vector of marker symbols to be remapped.
`report_file`	The path to a MRK_List1.rpt file. Leave this as `NULL` and the function will automatically download the report from `https://www.informatics.jax.org/downloads/reports/MRK_List1.rpt`.
`n_max`	Maximum number of lines to read from the `report_file`.

Value

A character vector of most up to date symbols.

Examples

```
rpt_ex01 <- report_example("MRK_List1-EX01.rpt")
read_report(report_file = rpt_ex01, report_type = "MRK_List1") |>
  dplyr::select("marker_status", "marker_symbol", "marker_symbol_now")

# NB:
#   - "1700024N20Rik" has two conflicting mappings, so maps to `NA`.
#   - "Hes1" is not present in MRK_List1-EX01.rpt, so maps to `NA`.
#   - "Plpbp" (official) and "Prosc" (withdrawn) both map to "Plpbp"

marker_symbols <- c("2200002F22Rik", "Plpbp", "Prosc", "1700024N20Rik", "Hes1")
symbol_to_symbol(x = marker_symbols, report_file = rpt_ex01)
```

Index