Package 'stoRy'

April 9, 2023

Title Download, Explore, and Analyze Literary Theme Ontology Data

Version 0.2.1

Date 2023-04-08

Description Download, explore, and analyze Literary Theme Ontology themes and thematically annotated story data. To learn more about the project visit <https://github.com/theme-ontology/theming> and <https://www.themeontology.org>.

License GPL-3

URL https://github.com/theme-ontology/stoRy, https://github.com/theme-ontology/theming, https://www.themeontology.org/

BugReports https://github.com/theme-ontology/stoRy/issues

Depends R (>= 3.5.0)

Imports cli, crayon, dplyr, fansi, httr, lifecycle, purrr, R6, rappdirs, readr, rlang, stringr, tibble, tidyjson, tidyr, utils

Suggests covr, curl, isa2, jsonlite, knitr, progress, rmarkdown, testthat

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Paul Sheridan [aut, cre] (<https://orcid.org/0000-0002-5484-1951>), Oshan Modi [aut], Mikael Onsjö [aut]

Maintainer Paul Sheridan <paul.sheridan.stats@gmail.com>

Repository CRAN

Date/Publication 2023-04-09 16:30:02 UTC

R topics documented:

`stoRy-package` *stoRy: Download, Explore, and Analyze Literary Theme Ontology Data*

Description

Download, explore, and analyze Literary Theme Ontology themes and thematically annotated story data. To learn more about the project visit https://github.com/theme-ontology/theming and https://www.themeontology.org.

Details

[Stable]

The **stoRy** package provides utilities for working with LTO data. The LTO is a hierarchically organized collection of carefully defined "themes" that can be expected to arise in multiple "stories" (i.e. works of fiction). Included in the package are functions to download and cache LTO data, explore LTO themes and thematically annotated stories, and analyze the thematically annotated story data in interesting ways.

General resources:

- stoRy package GitHub repository: https://github.com/theme-ontology/stoRy
- LTO project website: https://www.themeontology.org
- LTO project GitHub repositories: https://github.com/theme-ontology
- LTO conference paper in *Proceedings of the Joint Ontology Workshops 2019 Episode V: The Styrian Autumn of Ontology*

Author(s)

Maintainer: Paul Sheridan <paul.sheridan.stats@gmail.com> (ORCID)

Authors:

- Oshan Modi
- Mikael Onsjö

See Also

Useful links:

- https://github.com/theme-ontology/stoRy
- https://github.com/theme-ontology/theming
- https://www.themeontology.org/
- Report bugs at https://github.com/theme-ontology/stoRy/issues

---

`clone-lto` *Clone LTO data*

Description

[Maturing]

Clone internally stored active LTO version data.

`clone_active_themes_tbl()` returns a tibble of active LTO version themes.

`clone_active_stories_tbl()` returns a tibble of active LTO version stories.

`clone_active_collections_tbl()` returns a tibble of active LTO version collections.

`clone_active_metadata_tbl()` returns a tibble of active LTO version metadata.

Usage

```
clone_active_themes_tbl()

clone_active_stories_tbl()

clone_active_collections_tbl()

clone_active_metadata_tbl()
```

See Also

Run lto-demo to view the LTO demo data help page.

Run lto to find out how to load LTO versions.

Examples

```
## Not run:
# Make copies of the LTO demo data:
set_lto("demo")
themes_tbl <- clone_active_themes_tbl()
stories_tbl <- clone_active_stories_tbl()
collections_tbl <- clone_active_collections_tbl()
metadata_tbl <- clone_metadatae_stories_tbl()

## End(Not run)
```

`Collection` *R6 class representing a collection of LTO thematically annotated stories*

Description

[Maturing]

The **stoRy** package uses the `Collection` R6 class to represent a set of related LTO thematically annotated stories. This class is mostly useful for accessing information about a collection of stories for which the collection ID is known in advance.

Details

The class operates on the story collection of whichever LTO version happens to be actively loaded into the **stoRy** package level environment. This is the LTO `demo` version by default. Run `which_lto()` to check which LTO version is active in your R session.

Search the latest LTO `dev` version collections on the Theme Ontology website at https://www.themeontology.org/stories.

Alternatively, it is possible to read in a user-defined collection from file. In this case, the collection ID as defined in the file must match the `collection_id` input parameter.

Methods

Public methods:

- `Collection$new()`
- `Collection$collection_id()`
- `Collection$title()`
- `Collection$description()`
- `Collection$date()`
- `Collection$references()`
- `Collection$component_story_ids()`
- `Collection$themes()`
- `Collection$source()`
- `Collection$size()`

- Collection$obj_internal_tbl()
- Collection$print()
- Collection$clone()

Method `new()`**:** Initialize a collection of LTO thematically annotated stories.

Usage:

```
Collection$new(collection_id, file = NULL, verbose = TRUE)
```

Arguments:

`collection_id` A length-one character vector corresponding to the ID of an LTO collection of stories.

`file` A file name of a collection file or path to a collection file or a string. Files must end with the standard .st.txt extension used for story and collection files.
If `file` is a file name, then the file is assumed to reside in the current working directory.

`verbose` A logical value indicating whether status messages should be output to console.

Returns: A new `Collection` object.

Method `collection_id()`**:** return A length-one character vector corresponding to the collection ID.

Usage:

```
Collection$collection_id()
```

Method `title()`**:** return A length-one character vector corresponding to the collection title.

Usage:

```
Collection$title()
```

Method `description()`**:** return A length-one character vector of collection defining text.

Usage:

```
Collection$description()
```

Method `date()`**:** return A length-one character vector typically of the form "yyyy-yyyy" indicating the start and end year for stories in the collection.

Usage:

```
Collection$date()
```

Method `references()`**:** return A tibble of collection reference urls, if any.

Usage:

```
Collection$references()
```

Method `component_story_ids()`**:** return A tibble of member story IDs.

Usage:

```
Collection$component_story_ids()
```

Method `themes()`**:** return A tibble of thematic annotations.

Usage:

```
Collection$themes()
```

Method `source()`**:** return The path of the st.txt collection file. This is the file path as it occurs on the Theme Ontology GitHub repository at https://github.com/theme-ontology/theming.

Usage:

```
Collection$source()
```

Method `size()`**:** return A length-one numeric vector containing the number of stories in the collection.

Usage:

```
Collection$size()
```

Method `obj_internal_tbl()`**:** return A special tibble that is used internally by package functions.

Usage:

```
Collection$obj_internal_tbl()
```

Method `print()`**:** Print collection object info to console.

Usage:

```
Collection$print(canonical = FALSE, n = NULL, width = NULL, ...)
```

Arguments:

`canonical` Set to FALSE for pretty output.

`n` Maximum number of component story IDs to print to console. This defaults to NULL which means the `stoRy_opt("print_min")` value is used. Run `options(stoRy.print_min = 25L)` to set the minimum number of printed component story IDs to be 25. Run `stoRy_opt("print_max")` to check the maximum number of stories that can be printed to console. This value can be changed in the same way as with `stoRy.print_min`.

`width` Width of text output to generate. This defaults to NULL, which means the `stoRy_opt("width")` value is used. Run `options(stoRy.width = 120L)` to change the column width to be 120 characters, etc.

`...` Additional arguments

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

Use `Story()` to initialize an LTO thematically annotated story.

Use `Theme()` to initialize an LTO theme.

Use `Themeset()` to initialize a set of related LTO themes.

Examples

```
## Not run:
# Initialize a collection:
set_lto("demo")
collection <- Collection$new(collection_id = "Collection: tvseries: The Twilight Zone (1959)")

# Print collection info to console:
collection

# Print collection info in canonical st.txt format:
collection$print(canonical = TRUE)

# Initialize a collection from file:
set_lto("demo")
file <- system.file("extdata/rolling-stone-best-ttz1959-episodes.st.txt", package = "stoRy")
collection_id <- "Collection: Rolling Stone 25 Best Twilight Zone Original Series Episodes"
collection <- Collection$new(collection_id, file)
collection

# Initialize a collection from a string:
set_lto("demo")
file <- I("Collection: Rolling Stone 25 Best Twilight Zone Original Series Episodes
==========================================================================

:: Title
Rolling Stone 25 Best Twilight Zone Original Series Episodes

:: Date
1959-1964

:: Description
Rolling Stone Magazine's list of the 25 best episodes from the original
Twilight Zone anthology television series, which ran for five seasons on CBS
from 1959 to 1964, as compiled by David Fear, Sean T. Collins, and Angie
Martoccio.

:: References
https://www.rollingstone.com/tv/tv-features/25-best-twilight-zone-episodes-list-812043/

:: Collections
Collection: Rolling Stone 25 Best Twilight Zone Original Series Episodes

:: Component Stories
tz1959e3x24
tz1959e1x22
tz1959e2x06
tz1959e5x03
tz1959e2x15
tz1959e2x28
tz1959e1x08
tz1959e3x14
tz1959e3x05
```

```
tz1959e5x06
tz1959e3x08
tz1959e1x01
tz1959e1x21
tz1959e1x34
tz1959e2x07
tz1959e1x13
tz1959e1x09
tz1959e3x10
tz1959e1x16
tz1959e1x28
tz1959e1x30
tz1959e3x33
tz1959e3x01
tz1959e2x22
tz1959e5x25")
collection_id <- unlist(strsplit(file, split = "\n"))[1]
collection <- Collection$new(collection_id, file)
collection

## End(Not run)
```

`get_enriched_themes` *Find over-represented themes in a collection*

Description

[Maturing]

`get_enriched_themes()` calculates the top `m` most over-represented (or enriched) themes in a sub-collection of interest from a background collection.

Usage

```
get_enriched_themes(
  test_collection,
  background_collection = NULL,
  top_m = 10,
  weights = list(choice = 3, major = 2, minor = 1),
  explicit = TRUE,
  min_freq = 1,
  blacklist = NULL,
  metric = c("hgt", "tfidf")
)
```

Arguments

`test_collection`
: A `Collection()` class object of stories to assay for over-represented themes.

`background_collection`
: A `Collection()` class object the stories of are a superset of the `test_collection` stories.

  If `NULL`, the collection of all stories in the actively loaded LTO version is used.

`top_m`
: Maximum number of themes to report. The default is `top_m=10`.

  If `Inf`, all themes occurring at least `min_occurrence` times in the collection are reported.

`weights`
: A list assigning nonnegative weights to choice, major, and minor theme levels. The default weighting `list(choice = 3, major = 2, minor = 1)` counts each *choice* usage three times, each *major* theme usage twice, and each *minor* theme usage once. Use the uniform weighting `list(choice = 1, major = 1, minor = 1)` weights theme usages equally regardless of level. At least one weight must be positive.

`explicit`
: Set to `FALSE` to include ancestor themes of the explicit thematic annotations.

`min_freq`
: Drop themes occurring less than this number of times from the analysis. The default `min_freq=1` results in no themes are discarded.

`blacklist`
: A `Themeset()` class object. A themeset containing themes to be dropped from the analysis.

  If `NULL`, no themes are filtered.

`metric`
: A character vector specifying the choice of scoring function. Use `metric = "hgt"` for the *hypergeometric test*, and `metric = "tfidf"` for *term frequency-inverse document frequency*. The default specification of `metric = c("hgt", "tfidf")` results in the hypergeometric test being used in the analysis.

Details

The test collection of `n` stories, $S[1], \ldots, S[n]$, is represented as a weighted bag-of-words, where each *choice* theme in story $S[j] (j = 1, \ldots, n)$ is counted `weights$choice` times, each *major* theme `weights$major` times, and each *minor* theme `weights$choice` times.

The background collection of `N` stories, $S[1], \ldots, S[N]$, is a superset of the test collection that is likewise represented as a weighted bag-of-words.

Theme enrichment scores are calculated according to the hypergeometric test by default. Set `metric = "tfidf"` to use TF-IDF weights for the enrichment scores.

Value

Returns a `tibble` with `top_m` rows (themes) and 10 columns:

`theme_name`:	`m`-th most over-represented theme in the test collection
`k`:	Number of test collection stories featuring the theme
`k_bar`:	Weighted counts of the theme summed over the test collection stories
`n`:	Number of stories in the test collection
`n_bar`:	Sum of all weighted counts of test collection themes
`K`:	Number of background collection stories featuring the theme
`K_bar`:	Weighted counts of the theme summed over the background collection stories
`N`:	Number of stories in the background collection

N_bar:	Sum of all weighted counts of background collection themes
score:	Either the negative base 10 logarithm of the Hypergeometric test (if metric = "hgt") or TF-IDF (if metric = '

References

Mikael Onsjö, Paul Sheridan (2020). Theme Enrichment Analysis: A Statistical Test for Identifying Significantly Enriched Themes in a List of Stories with an Application to the Star Trek Television Franchise. *Digital Studies/le Champ Numérique*, 10(1), 1. DOI: doi:10.16995/dscn.316

Examples

```
## Not run:
# Retrieve the top 10 most enriched themes in "The Twilight Zone" (1959)
# series episodes with all demo version stories as background:
set_lto("demo")
test_collection <- Collection$new(collection_id = "Collection: tvseries: The Twilight Zone (1959)")
result_tbl <- get_enriched_themes(test_collection)
result_tbl

# Run the same analysis on "The Twilight Zone" (1959) series without
# including minor level themes:
result_tbl <- get_enriched_themes(test_collection, weights = list(choice = 1, major = 1, minor = 0))
result_tbl

## End(Not run)
```

---

get_featured_themes *Find the most frequently occurring themes in a collection*

Description

[Maturing]

`get_featured_themes()` calculates the top `m` most frequently occurring themes in a collection.

Usage

```
get_featured_themes(
  collection = NULL,
  top_m = 10,
  weights = list(choice = 3, major = 2, minor = 1),
  explicit = TRUE,
  min_freq = 1,
  blacklist = NULL
)
```

Arguments

collection	A Collection() class object. If NULL, the collection of all stories in the actively loaded LTO version is used.
top_m	Maximum number of themes to report. The default is top_m=10. If Inf, all themes occurring at least min_occurrence times in the collection are reported.
weights	A list assigning nonnegative weights to choice, major, and minor theme levels. The default weighting list(choice = 3, major = 2, minor = 1) counts each *choice* usage three times, each *major* theme usage twice, and each *minor* theme usage once. Use the uniform weighting list(choice = 1, major = 1, minor = 1) weights theme usages equally regardless of level. At least one weight must be positive.
explicit	Set to FALSE to include ancestor themes of the explicit thematic annotations.
min_freq	Drop themes occurring less than this number of times from the analysis. The default min_freq=1 results in no themes are discarded.
blacklist	A Themeset() class object. A themeset containing themes to be dropped from the analysis. If NULL, no themes are filtered.

Details

The input collection of n stories, $S[1], \ldots, S[n]$, is represented as a weighted bag-of-words, where each *choice* theme in story $S[j] (j = 1, \ldots, n)$ is counted weights$choice times, each *major* theme weights$major times, and each *minor* theme weights$choice times.

Value

Returns a tibble with top_m rows (themes) and 6 columns:

theme_name:	m-th most frequently occurring theme in the collection
k:	Number of collection stories featuring the theme
k_bar:	Weighted counts of the theme summed over the collection stories
n:	Number of stories in the collection
n_bar:	Sum of all weighted counts of collection themes
tp:	Theme weighted term proportion (i.e. k_bar/n_bar)

Examples

```
## Not run:
# Retrieve the top 10 most featured themes in "The Twilight Zone" franchise
# stories:
set_lto("demo")
result_tbl <- get_featured_themes()
result_tbl

# Retrieve the top 10 most featured themes in "The Twilight Zone" franchise
```

```
# stories not including any minor level themes:
set_lto("demo")
result_tbl <- get_featured_themes(weights = list(choice = 1, major = 1, minor = 0))
result_tbl

# Retrieve the top 10 most featured themes in "The Twilight Zone" (1959)
# television series episodes:
collection <- Collection$new(collection_id = "Collection: tvseries: The Twilight Zone (1959)")
result_tbl <- get_featured_themes(collection)
result_tbl

## End(Not run)
```

`get_similar_stories` *Find similar stories to a given story*

Description

[Maturing]

`get_similar_stories` calculates the top `n` most thematically similar stories to a given story.

Usage

```
get_similar_stories(
  query_story,
  background_collection = NULL,
  top_n = 10,
  weights = list(choice = 3, major = 2, minor = 1),
  explicit = TRUE,
  min_freq = 1,
  blacklist = NULL,
  metric = c("hgt", "tfidf")
)
```

Arguments

`query_story`	A `Story()` class object defining a story of interest. Thematically similar stories to this one will be returned.
`background_collection`	A `Collection()` class object the stories in which to search for similar stories to the `query_story` input story. If `NULL`, the collection of all stories in the actively loaded LTO version is used.
`top_n`	Maximum number of similar stories to report. The default is `top_n=10`. If `Inf`, all stories in the background collection are reported.

weights	A list assigning nonnegative weights to choice, major, and minor theme levels. The default weighting `list(choice = 3, major = 2, minor = 1)` counts each *choice* usage three times, each *major* theme usage twice, and each *minor* theme usage once. Use the uniform weighting `list(choice = 1, major = 1, minor = 1)` weights theme usages equally regardless of level. At least one weight must be positive.
explicit	Set to `FALSE` to include ancestor themes of the explicit thematic annotations.
min_freq	Drop themes occurring less than this number of times from the analysis. The default `min_freq=1` results in no themes are discarded.
blacklist	A `Themeset()` class object. A themeset containing themes to be dropped from the analysis. If `NULL`, no themes are filtered.
metric	A character vector specifying the choice of weighting to use in the *cosine similarity* measure used to evaluate story thematic similarity. Use `metric = "hgt"` for hypergeometric test P-value weights and `metric = "tfidf"` for TF-IDF weights. The default specification of `metric = c("hgt", "tfidf")` results in hypergeometric test P-values being used as weights.

Value

Returns a `tibble` with `top_n` rows (stories) and 5 columns:

story_id:	n-th most thematically similar story to the query story
title:	Reference story title
description:	Reference story description
score:	Cosine similarity score with hypergeometric test weights (if `metric = "hgt"`) or TF-IDF weights (if `metri`
common_themes:	List of themes common to both the query and reference story

References

Paul Sheridan, Mikael Onsjö, Claudia Becerra, Sergio Jimenez, Georg Dueñas (2019). *An Ontology-Based Recommender System with an Application to the Star Trek Television Franchise*. *Future Internet*, 11(9), 182. DOI: doi:10.3390/fi11090182

Examples

```
## Not run:
# Retrieve the top 10 most similar stories to the classic "The Twilight
# Zone" series episode "Nightmare at 20,000 Feet" (1959):
set_lto("demo")
query_story <- Story$new(story_id = "tz1959e5x03")
result_tbl <- get_similar_stories(query_story)
result_tbl

# Retrieve the top 10 most similar stories to the classic "The Twilight
# Zone" series episode "Nightmare at 20,000 Feet" (1959) without taking
# minor themes into account:
```

```
set_lto("demo")
query_story <- Story$new(story_id = "tz1959e5x03")
result_tbl <- get_similar_stories(query_story, weights = list(choice = 3, major = 2, minor = 0))
result_tbl

# Retrieve the top 10 most similar stories to the classic "The Twilight
# Zone" series episode "Nightmare at 20,000 Feet" (1959) when implicitly
# featured themes are included in the similarity calculation:
set_lto("demo")
query_story <- Story$new(story_id = "tz1959e5x03")
result_tbl <- get_similar_stories(query_story, explicit = FALSE)
result_tbl

## End(Not run)
```

`get_story_clusters` *Find clusters of similar stories*

Description

[Maturing]

`get_story_clusters` classifies the stories in a collection according to thematic similarity.

Usage

```
get_story_clusters(
  collection = NULL,
  weights = list(choice = 3, major = 2, minor = 1),
  explicit = TRUE,
  min_freq = 1,
  min_size = 3,
  blacklist = NULL
)
```

Arguments

`collection`	A `Collection()` class object. If `NULL`, the collection of all stories in the actively loaded LTO version is used.
`weights`	A list assigning nonnegative weights to choice, major, and minor theme levels. The default weighting `list(choice = 3, major = 2, minor = 1)` counts each *choice* usage three times, each *major* theme usage twice, and each *minor* theme usage once. Use the uniform weighting `list(choice = 1, major = 1, minor = 1)` weights theme usages equally regardless of level. At least one weight must be positive.
`explicit`	Set to `FALSE` to include ancestor themes of the explicit thematic annotations.
`min_freq`	Drop themes occurring less than this number of times from the analysis. The default `min_freq=1` results in no themes are discarded.

`min_size`	Minimum cluster size. The default is `min_size=3`.
`blacklist`	A `Themeset()` class object. A themeset containing themes to be dropped from the analysis. If `NULL`, no themes are filtered.

Details

The input collection of `n` stories, $S[1], \ldots, S[n]$, is represented as a weighted bag-of-words, where each *choice* theme in story $S[j] (j = 1, \ldots, n)$ is counted `weights$choice` times, each *major* theme `weights$major` times, and each *minor* theme `weights$choice` times.

The function classifies the stories according to thematic similarity using the Iterative Signature Algorithm (ISA) biclustering algorithm as implemented in the `isa2` R package. The clusters are "soft" meaning that a story can appear in multiple clusters.

Install `isa2` package by running the command `install.packages(\"isa2\")` before calling this function.

Value

Returns a `tibble` with `r` rows (story clusters) and 4 columns:

`cluster_id`:	Story cluster integer ID
`stories`:	A tibble of stories comprising the cluster
`themes`:	A tibble of themes common to the clustered stories
`size`:	Number of stories in the cluster

References

Gábor Csárdi, Zoltán Kutalik, Sven Bergmann (2010). Modular analysis of gene expression data with R. *Bioinformatics*, 26, 1376-7.

Sven Bergmann, Jan Ihmels, Naama Barkai (2003). Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical Review E*, 67, 031902.

Gábor Csárdi (2017). isa2: The Iterative Signature Algorithm. R package version 0.3.5. https://cran.r-project.org/package=isa2

Examples

```
## Not run:
# Cluster "The Twilight Zone" franchise stories according to thematic
# similarity:
library(dplyr)
set_lto("demo")
set.seed(123)
result_tbl <- get_story_clusters()
result_tbl

# Explore a cluster of stories related to traveling back in time:
cluster_id <- 3
pull(result_tbl, stories)[[cluster_id]]
```

```
pull(result_tbl, themes)[[cluster_id]]

# Explore a cluster of stories related to mass panics:
cluster_id <- 5
pull(result_tbl, stories)[[cluster_id]]
pull(result_tbl, themes)[[cluster_id]]

# Explore a cluster of stories related to executions:
cluster_id <- 7
pull(result_tbl, stories)[[cluster_id]]
pull(result_tbl, themes)[[cluster_id]]

# Explore a cluster of stories related to space aliens:
cluster_id <- 10
pull(result_tbl, stories)[[cluster_id]]
pull(result_tbl, themes)[[cluster_id]]

# Explore a cluster of stories related to old people wanting to be young:
cluster_id <- 11
pull(result_tbl, stories)[[cluster_id]]
pull(result_tbl, themes)[[cluster_id]]

# Explore a cluster of stories related to wish making:
cluster_id <- 13
pull(result_tbl, stories)[[cluster_id]]
pull(result_tbl, themes)[[cluster_id]]

## End(Not run)
```

`lto` *Work with LTO versions*

Description

[Maturing]

Download, configure, view metadata about, and navigate among different LTO versions.

`which_lto()` returns a length-one character vector corresponding to the active LTO version. This is the version that is loaded into the stoRy package environment. It is the `demo` version by default.

`print_lto()` prints basic LTO version metadata to console.

`fetch_lto_version_tags()` returns a character vector consisting of all exiting LTO version tags. The LTO versioned release tags are downloaded from the Theme Ontology GitHub website at https://github.com/theme-ontology/theming/releases.

`lto_version_statuses()` prints to console the status (available for download/cached/defunct) for each LTO version.

`configure_lto()` downloads and configures LTO version releases hosted on the Theme Ontology website at https://www.themeontology.org/data.

`set_lto()` sets an LTO version as the active version. This means that package functions will act on this version. The active version is set to `demo` by default.

Usage

```
which_lto()

print_lto()

fetch_lto_version_tags(verbose = TRUE)

lto_version_statuses(verbose = TRUE)

configure_lto(
  version,
  verbose = TRUE,
  overwrite_json = FALSE,
  overwrite_rds = FALSE
)

set_lto(version, verbose = TRUE, load_background_collection = TRUE)
```

Arguments

verbose	A logical value indicating whether status messages should be output to console.
version	A length-one character vector specifying an LTO version tag. Set to "latest" to configure the latest numbered version, and "dev" to configure the LTO developmental version.
overwrite_json	A logical value indicating whether previously downloaded JSON files (if any) should be re-downloaded and overwritten.
overwrite_rds	A logical value indicating whether previously generated Rds files (if any) should be regenerated and overwritten.
load_background_collection	**[Experimental]** A logical value indicating whether the default background collection of all LTO thematically annotated stories should be loaded into the stoRy package environment when activating an LTO version. Setting this to FALSE may result in some analysis functions not working.

References

Paul Sheridan, Mikael Onsjö, and Janna Hastings, The Literary Theme Ontology for Media Annotation and Information Retrieval, Proceedings of JOWO2019: The Joint Ontology Workshops, Graz, Austria, September 22-26 (https://ceur-ws.org/Vol-2518/paper-WODHSA8.pdf).

See Also

Run lto-demo to view the LTO demo data help page.

Run print_stoRy_cache_info() to list all cached files.

Examples

```
## Not run:
# Check which LTO version is active:
which_lto()

# Print summary info about active LTO version to console:
print_lto()

# Print summary of existing LTO versions:
fetch_lto_version_tags()

# Store LTO version tags as a character vector:
lto_version_tags <- fetch_lto_version_tags()
lto_version_tags

# Configure the latest LTO version, only if it is not already setup:
configure_lto(version = "latest")

# Reconfigure the latest LTO version from scratch:
configure_lto(version = "latest", overwrite_json = TRUE, overwrite_rds = TRUE)

# Change to latest LTO version:
set_lto(version = "latest")

## End(Not run)
```

`lto-demo` *LTO demo data*

Description

A family of datasets extracted from LTO v0.3.3, comprising 2872 LTO themes and 335 thematically annotated *The Twilight Zone* American media franchise stories.

Found in the data are thematic annotations for

- 156 *The Twilight Zone* (1959) television series episodes
- 3 *Twilight Zone: The Movie* (1983) film sub-stories
- 110 *The Twilight Zone* (1985) television series episodes
- 3 *Twilight Zone: Rod Serling's Lost Classics* (1994) film sub-stories
- 43 *The Twilight Zone* (2002) television series episodes
- 20 *The Twilight Zone* (2019) television series episodes

Format

The data consists of four `tibble` class objects:

1. `metadata_tbl`: a `tibble` of LTO demo data summary information with 7 rows and 2 columns:

name:	Metadatum name (e.g. version, encoding, etc.)
value:	Metadatum value (e.g. "demo", "UTF-8", etc.)

2. `themes_tbl`: a `tibble` of LTO demo data themes with 2872 rows (themes) and 11 columns:

theme_index:	Integer ID (unique)
theme_name:	Theme name (unique)
description:	Theme definition
notes:	Theme definition elaborations and caveats
aliases:	Theme name aliases
template:	Theme special cases
parents:	Parent themes
ancestors:	Ancestor themes
examples:	Example usages
references:	Reference URLs
source:	Path to file where theme is defined

3. `stories_tbl`: a `tibble` of LTO demo data stories with 335 rows (stories) and 10 columns:

story_index:	Integer ID (unique)
story_id:	Story ID (unique)
title:	Official title
date:	Release date
description:	Information used for identifying the story
component_story_ids:	Sub-story story IDs (used for frame stories)
collections:	Collection IDs of collections to which the story belongs
references:	Reference URLs
themes:	Thematic annotations
source:	Path to file where story is defined

4. `collections_tbl`: a `tibble` of LTO demo data collections with 5 rows (story collections) and 9 columns:

collection_index:	Integer ID (unique)
collection_id:	Collection ID (unique)
title:	The collection ID stripped of all colon separated prefixes
date:	Earliest and latest dates of stories in the collection
description:	Minimal information defining the collection
component_story_ids:	Story IDs of member stories of the collection
references:	Reference URLs
themes:	Collection level thematic annotations
source:	Path to file where collection is defined

Note

The data is stored internally in the package 'R/sysdata.rda' file and cannot be accessed directly. Check the examples below for more on how to best explore the data.

Source

Theme Ontology. (2021, October 31). LTO v0.3.3. https://github.com/theme-ontology/theming/releases/tag/v0.3.3

References

The Twilight Zone. (2021, July 25). In Wikipedia https://en.wikipedia.org/wiki/The_Twilight_Zone

Examples

```
# Print a copy of LTO demo version metadata to console:
set_lto("demo")
demo_metadata_tbl <- clone_active_metadata_tbl()
demo_metadata_tbl

# Print a copy of LTO demo themes to console:
set_lto("demo")
demo_themes_tbl <- clone_active_themes_tbl()
demo_themes_tbl

# Print a copy of LTO demo stories to console:
set_lto("demo")
demo_stories_tbl <- clone_active_stories_tbl()
demo_stories_tbl

# Print a copy of LTO demo collections to console:
set_lto("demo")
demo_collections_tbl <- clone_active_collections_tbl()
demo_collections_tbl
# Print collection IDs to console:
demo_collections_tbl$collection_id
# Print The Twilight Zone (2019) component story IDs to console:
library(dplyr)
collection_id <- "Collection: tvseries: The Twilight Zone (2019)"
demo_collections_tbl %>%
  filter(collection_id %in% !!collection_id) %>%
  pull(component_story_ids) %>%
  unlist(use.names = FALSE)
```

`Story` *R6 class representing an LTO thematically annotated story*

Description

[Maturing]

The **stoRy** package uses the `Story` R6 class to represent the LTO thematic annotations for individual works of fiction. This class is mostly useful for accessing information about an LTO thematically annotated story for which the story ID is known in advance.

Details

The class operates on the stories of whichever LTO version happens to be actively loaded into the **stoRy** package level environment. This is the LTO `demo` version by default. Run `which_lto()` to check which LTO version is active in your R session.

Search the latest LTO dev version stories on the Theme Ontology website at https://www.themeontology.org/stories.

Methods

Public methods:

- `Story$new()`
- `Story$story_id()`
- `Story$title()`
- `Story$description()`
- `Story$date()`
- `Story$references()`
- `Story$collections()`
- `Story$themes()`
- `Story$source()`
- `Story$obj_internal_tbl()`
- `Story$print()`
- `Story$clone()`

Method `new()`**:** Initialize an LTO thematically annotated story.

Usage:

```
Story$new(story_id)
```

Arguments:

`story_id` A length-one character vector corresponding to the ID of an LTO thematically annotated work of fiction.

Returns: A new `Story` object.

Method `story_id()`**:** return A length-one character vector corresponding to the story ID.

Usage:

```
Story$story_id()
```

Method `title()`**:** return A length-one character vector corresponding to the story title.

Usage:

```
Story$title()
```

Method `description()`**:** return A length-one character vector corresponding to some summary information about the story. This is typically a synopsis and/or details about the authorship, production, distribution, etc.

Usage:

```
Story$description()
```

Method `date()`**:** return A length-one character vector corresponding to the story release date.

Usage:

```
Story$date()
```

Method `references()`**:** return A tibble of story reference urls, if any.

Usage:

```
Story$references()
```

Method `collections()`**:** return A tibble of LTO collections to which the story belongs, if any.

Usage:

```
Story$collections()
```

Method `themes()`**:** return A tibble of thematic annotations.

Usage:

```
Story$themes()
```

Method `source()`**:** return The path of the st.txt file containing the story thematic annotations. This is the file path as it occurs on the Theme Ontology GitHub repository at https://github.com/theme-ontology/theming.

Usage:

```
Story$source()
```

Method `obj_internal_tbl()`**:** return A special tibble that is used internally by package functions.

Usage:

```
Story$obj_internal_tbl()
```

Method `print()`**:** Print story object info to console.

Usage:

```
Story$print(canonical = FALSE, width = NULL, ...)
```

Arguments:

`canonical` Set to FALSE for pretty output.

`width` Width of text output to generate. This defaults to NULL, which means the `stoRy_opt("width")` value is used. Run `options(stoRy.width = 120L)` to change the column width to be 120 characters, etc.

`...` Additional arguments

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

Use `Collection()` to initialize an collection of LTO thematically annotated stories.

Use `Theme()` to initialize an LTO theme.

Use `Themeset()` to initialize a set of related LTO themes.

Examples

```
## Not run:
# Initialize the LTO `demo` version of a classic The Twilight Zone (1959) story:
set_lto("demo")
story <- Story$new(story_id = "tz1959e1x22")

# Print story info and thematic annotations to console:
story

# Print story info and thematic annotations in st.txt format:
story$print(canonical = TRUE)

# Return the story title:
story$title()

# Return the story description:
story$description()

# Return a tibble of thematic annotations:
story$themes()

## End(Not run)
```

`stoRy_cache` *Manage cached files*

Description

Manage **stoRy** package cached files.

`stoRy_cache_path()` returns the file path where **stoRy** package files are cached.

`stoRy_cache_details()` returns a tibble of cached file names with accompanying metadata. File size is in MB.

`delete_lto_version_cached_files()` delete cached files associated with an LTO version.

`delete_stoRy_cached_files()` delete cached files. Takes a vector of file path strings as input.

`delete_all_cached_stoRy_files()` clear cache.

`print_stoRy_cache_info()` print to console cache contents.

Usage

```
stoRy_cache_path()

stoRy_cache_details()

delete_lto_version_cached_files(version, force = TRUE, verbose = TRUE)

delete_stoRy_cached_files(files, force = TRUE)

delete_all_cached_stoRy_files(force = TRUE)

print_stoRy_cache_info()
```

Arguments

`version`	A length-one character vector specifying an LTO version tag. Set to "latest" to configure the latest numbered version, and "dev" to configure the LTO developmental version.
`force`	Set to TRUE to force delete files.
`verbose`	A logical value indicating whether status messages should be output to console.
`files`	A list of file path strings to delete from cache.

Examples

```
## Not run:
# list files in cache
stoRy_list_cached_files()

# List info for all files
print_stoRy_cache_info()

# delete all files in cache
# stoRy_delete_all_cached_files()

## End(Not run)
```

stoRy_opt *Set* **stoRy** *package global options*

Description

`stoRy_opt()` sets **stoRy** package global options.

Usage

```
stoRy_opt(x)
```

Arguments

`x`	A character string holding an option name. The possible values are `"width"`, `"print_min"`, and `"print_max"`.

Details

The package options control the formatting of the `Story()`, `Theme()`, `Collection()`, and `Themeset()` R6 classes when printing to console. The options are as follows:

`width`:	The column width in characters of printed to console output (78 characters by default)
`print_min`:	The minimum number of entries to print to console (10 characters by default)
`print_max`:	The maximum number of entries to print to console (100 characters by default)

See Also

Use `Story()` to initialize an LTO thematically annotated story.

Use `Theme()` to initialize an LTO theme.

Use `Collection()` to initialize an collection of LTO thematically annotated stories.

Use `Themeset()` to initialize a set of related LTO themes.

Examples

```
## Not run:
# Check the current option values:
stoRy_opt("width")
stoRy_opt("print_min")
stoRy_opt("print_max")

# Set the column width to 120 characters:
options(stoRy.width = 120L)

# Set the minimum number of printed entries to be 25:
options(stoRy.print_min = 25L)

# Set the maximum number of printed entries to be 250:
```

```
options(stoRy.print_max = 250L)

## End(Not run)
```

Theme *R6 class representing an LTO literary theme*

Description

[Maturing]

The **stoRy** package uses the Theme R6 class to represent individual LTO literary themes. This class is mostly useful for accessing information about an LTO theme for which the theme name is known in advance.

Details

The class operates on the themes of whichever LTO version happens to be actively loaded into the **stoRy** package level environment. This is the LTO demo version by default. Run `which_lto()` to check which LTO version is active in your R session.

Search the latest LTO dev version themes on the Theme Ontology website at https://www.themeontology.org/themes.

Methods

Public methods:

- `Theme$new()`
- `Theme$theme_name()`
- `Theme$aliases()`
- `Theme$description()`
- `Theme$notes()`
- `Theme$references()`
- `Theme$examples()`
- `Theme$parents()`
- `Theme$ancestors()`
- `Theme$source()`
- `Theme$annotations()`
- `Theme$print()`
- `Theme$clone()`

Method `new()`**:** Initialize an LTO theme.

Usage:

```
Theme$new(theme_name)
```

Arguments:

`theme_name` A length-one character vector corresponding to an LTO theme name.

Returns: A new Theme object.

Method `theme_name()`**:** return A length-one character vector corresponding to the theme name.

Usage:

```
Theme$theme_name()
```

Method `aliases()`**:** return A tibble of theme name aliases.

Usage:

```
Theme$aliases()
```

Method `description()`**:** return A length-one character vector corresponding to the theme description.

Usage:

```
Theme$description()
```

Method `notes()`**:** return A tibble of caveats that accompany that theme description. This is empty for the vast majority of themes.

Usage:

```
Theme$notes()
```

Method `references()`**:** return A tibble of references.

Usage:

```
Theme$references()
```

Method `examples()`**:** return A tibble of example usages of the theme, if any.

Usage:

```
Theme$examples()
```

Method `parents()`**:** return A tibble of the theme's parent theme names.

Usage:

```
Theme$parents()
```

Method `ancestors()`**:** return A tibble of the theme's ancestor theme names.

Usage:

```
Theme$ancestors()
```

Method `source()`**:** return The path of the th.txt file containing the theme. This is the file path as it occurs on the Theme Ontology GitHub repository at https://github.com/theme-ontology/theming.

Usage:

```
Theme$source()
```

Method `annotations()`**:** return A tibble with one row for each story in which the theme is featured. The first column is the LTO story ID, the second is the release data, the third is the level (choice/major/minor) at which the theme is featured, and the fourth a justification for applying the theme. Each column is of type character.

Usage:

```
Theme$annotations()
```

Method `print()`**:** Print theme object info to console.

Usage:

```
Theme$print(canonical = FALSE, width = NULL, ...)
```

Arguments:

`canonical` Set to FALSE for pretty output.

`width` Width of text output to generate. This defaults to NULL, which means the `stoRy_opt("width")` value is used. Run `options(stoRy.width = 120L)` to change the column width to be 120 characters, etc.

`...` Additional arguments.

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

Use `Collection()` to initialize an collection of LTO thematically annotated stories.

Use `Story()` to initialize an LTO thematically annotated story.

Use `Themeset()` to initialize a set of related LTO themes.

Examples

```
## Not run:
# Initialize an LTO `demo` version theme pertaining to Martians:
set_lto("demo")
theme <- Theme$new(theme_name = "Martian extraterrestrial")

# Print theme info to console:
theme

# Print theme info to console in the canonical th.txt format:
theme$print(canonical = TRUE)

# Return the theme description:
theme$description()

# Return references associated with the description, if any:
theme$references()

# Return theme name aliases, if any:
theme$aliases()

# Return the theme's parent theme names:
```

```
theme$parents()

# Return the theme's ancestor theme names:
theme$ancestors()

# Return a tibble of thematically annotated stories:
theme$annotations()

## End(Not run)
```

`Themeset` *R6 class representing an LTO themeset*

Description

[Experimental]

The **stoRy** package uses the `Themeset` R6 class to represent user defined collections of themes. Themesets are currently in an experimental stage of development, but can be expected to become an integrate part of future **stoRy** package analysis functions.

Details

Various themesets are hosted on the themesets Theme Ontology GitHub repository https://github.com/theme-ontology/themesets.

Methods

Public methods:

- `Themeset$new()`
- `Themeset$themeset_id()`
- `Themeset$description()`
- `Themeset$component_theme_names()`
- `Themeset$size()`
- `Themeset$obj_internal_tbl()`
- `Themeset$print()`
- `Themeset$clone()`

Method `new()`**:** Initialize a collection of LTO themes.

Usage:

```
Themeset$new(file, verbose = TRUE)
```

Arguments:

`file` Either a file name, a path to a file, a url, or a single single must contain at least one newline to be recognized as such (as string as opposed to a path or url). Files must end with the standard .thset.txt extension used for themeset files.
If `file` is a file name, then the file is assumed to reside in the current working directory.

verbose A logical value indicating whether status messages should be output to console.

Returns: A new Themeset object.

Method `themeset_id()`**:** return A length-one character vector corresponding to the themeset ID.

Usage:

```
Themeset$themeset_id()
```

Method `description()`**:** return A length-one character vector corresponding to the themeset description.

Usage:

```
Themeset$description()
```

Method `component_theme_names()`**:** return A tibble of member themes.

Usage:

```
Themeset$component_theme_names()
```

Method `size()`**:** return A length-one numeric vector containing the number of themes in the themeset.

Usage:

```
Themeset$size()
```

Method `obj_internal_tbl()`**:** a pre-computed table used internally by package functions

Usage:

```
Themeset$obj_internal_tbl()
```

Method `print()`**:** Print collection object info to console.

Usage:

```
Themeset$print(canonical = FALSE, n = NULL, width = NULL, ...)
```

Arguments:

canonical Set to FALSE for pretty output.

n Maximum number of component theme names to print to console. This defaults to NULL which means the `getOption("stoRy.print_min")` value is used. Run `options(stoRy.print_min = 25L)` to set the minimum number of printed component theme names to be 25. Run `stoRy_opt("print_max")` to check the maximum number of themes that can be printed to console. This value can be changed in the same way as with `stoRy.print_min`.

width Width of text output to generate. This defaults to NULL, which means the `stoRy_opt("width")` value is used. Run `options(stoRy.width = 120L)` to change the column width to be 120 characters, etc.

... Additional arguments.

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

Use Collection() to initialize an collection of LTO thematically annotated stories.

Use Story() to initialize an LTO thematically annotated story.

Use Theme() to initialize an LTO theme.

Examples

```
## Not run:
# Initialize a themeset from file:
set_lto("demo")
file <- system.file("extdata/immortality.thset.txt", package = "stoRy")
themeset <- Themeset$new(file)

# Print themeset info to console:
themeset

#' # Read themeset from a url and print to console:
set_lto("demo")
file <- paste0(
  "https://raw.githubusercontent.com/theme-ontology/",
  "master/demo/immortality.thset.txt"
)
themeset <- Themeset$new(file)
themeset

# Initialize a themeset directly from a string and print to console:
set_lto("demo")
file <- I("Themeset: immortality
====================

:: Description
Themes related to people living on well beyond what is considered to be a
normal human lifespan.

:: Component Themes
immortality
the flip side of immortality
the quest for immortality")
themeset <- Themeset$new(file)
themeset

## End(Not run)
```

Index