Package 'lobbyR'

January 13, 2026

Type Package

Title Get Federal Lobbying Disclosures

Version 0.1.0

Depends R (>= 4.1.0)

Author Chris Cioffi [aut, cre],
Aarushi Sahejpal [aut, ctb]

Maintainer Chris Cioffi <chrisjcioffi@gmail.com>

Description Gives users seeking federal lobbying disclosures an easier way to query the API maintained by the Senate federal lobbying disclosures database <https://lda.senate.gov/api/redoc/v1/> to find out how much companies and other entities are spending to lobby Congress and the federal government. It allows for search terms such as keywords, time periods and entity names. It then attempts to clean, or at least flag, filings that could provide incorrect results when seeking to answer the question: How much is being spent on lobbying our Congress and the administration and what issues do they care about?

License LGPL (>= 3)

Encoding UTF-8

Suggests spelling

Language en-US

Imports dplyr, httr2, keyring, rlang, stringr, tidyr

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2026-01-13 18:30:02 UTC

Contents

`flag_client_registrant_conflict` *Identify and Resolve Potential Double-Counting of Client/Registrant*

Description

Flags instances where an entity filed as a registrant (which, if following the rules, should include all lobbying spending by that entity) but also appeared as a client in filings by outside lobbyists. This helps avoid double-counting lobbying expenses.

Usage

```
flag_client_registrant_conflict(
  dataframe_that_i_determine,
  flag_conflict = TRUE,
  clean_doublecounts = TRUE
)
```

Arguments

`dataframe_that_i_determine`	A data frame, typically the name of the output from [get_filings()], containing lobbying filings for a given query.
`flag_conflict`	Logical. If `TRUE` (default), the function flags filings that are likely duplicates as described above.
`clean_doublecounts`	Logical. If `TRUE` (default), the function automatically filters out cases that could lead to double-counting.

Details

This function identifies:

- Self-lobbying entities (where registrant and client names match), which should include all lobbying spending.
- Cases where the same entity appears as a client in filings by other lobbyists, which should be excluded from aggregate totals to avoid double-counting.

The function adds a flag column to the data frame, indicating whether each filing is a "Report of all entity's spending", "Likely part of separate entity's report", or "No entity's report detected". If `clean_doublecounts = TRUE`, filings that are likely to cause double-counting are removed.

Note: This function is not perfect and may miss some edge cases. Manual review is recommended for critical analyses.

Value

A data frame with a new flag column and, optionally, with potential double-counting cases filtered out.

See Also

[get_filings()] for retrieving lobbying data, and [flag_dupes] for identifying duplicate or amended filings.

Examples

```
## Not run:
# Flag and clean potential double-counting cases
conflict_removed_and_flagged <- flag_client_registrant_conflict(
  cleaner_df,
  flag_conflict = TRUE,
  clean_doublecounts = TRUE
)

# Only flag, do not remove
conflict_flagged_only <- flag_client_registrant_conflict(
  cleaner_df,
  flag_conflict = TRUE,
  clean_doublecounts = FALSE
)

## End(Not run)
```

`flag_dupes` *Flag and Clean Duplicate or Dubious Lobbying Filings*

Description

'flag_dupes()'Identifies and flags potentially problematic or duplicate lobbying filings in a data frame (typically one returned by `get_filings()`). The function adds several diagnostic columns to help users spot filings that may require closer inspection, and can optionally remove all but the latest filing for each egistrant-client-quarter group.

Usage

```
flag_dupes(
  cleaned_dataframe_from_previous_function,
  find_duplicates = TRUE,
  attempt_cleaning = TRUE
)
```

Arguments

`cleaned_dataframe_from_previous_function`
: - A data frame, typically the output of `get_filings()`, containing lobbying filings to be checked for duplicates or other issues.

`find_duplicates`
: Logical. If `TRUE` (default), the function flags dubious filings using several heuristics and regex patterns.

`attempt_cleaning`
: Logical. If `TRUE` (default), the function removes all but the latest filing for each registrant-client-quarter group, assuming the most recent filing is the most accurate.

Details

The function creates several columns to help identify filings that may be of concern:

- `registration_or_termination`: TRUE if the filing is a registration or termination filing (detected via regex).
- `quarter_number`: Extracted quarter number from `filing_type`.
- `is_amendment`: TRUE if the filing is an amendment.
- `has_quarter`, `has_amendment`, `registration_termination`, `is_duplicate`: Various flags for duplicate or suspicious filings.
- `checkme`: "CHECK" if the row is flagged as potentially problematic, otherwise "PASS CHECK".

If `attempt_cleaning = TRUE`, the function keeps only the latest filing (by `dt_posted`) for each registrant, client, year, and filing period, after removing registration and termination filings.

Value

A data frame with additional columns indicating potential issues, and (optionally) with duplicate filings removed.

See Also

[get_filings()] for retrieving data from the API, and [flag_client_registrant_conflict] for methods to prevent doublecounting when entities that file lobbying disclosures as registrants, but pay outside lobbying firms too, also show up as clients.

Examples

```
## Not run:
# Flag and clean duplicate filings in a lobbying data frame
dupes_flag_test <- flag_dupes(
  df,
  find_duplicates = TRUE,
  attempt_cleaning = TRUE
)

# Only flag, do not remove duplicates
```

```
flagged_only <- flag_dupes(
  df,
  find_duplicates = TRUE,
  attempt_cleaning = FALSE
)

## End(Not run)
```

`get_filings` *Query Federal Lobbying Disclosures*

Description

'get_filings()' allows users to search the Senate federal lobbying disclosures database (https://lda.senate.gov/api/redoc/v1/) to ascertain much companies are spending to lobby the federal government or how much lobbyists are being paid on behalf of clients to lobby on various issues. The query searches based on various criteria including issues, filing periods, client names, registrant names, and dates.

Usage

```
get_filings(
  issues = c(""),
  issue_joiner = "",
  year = "",
  filing_period = "",
  client_name = "",
  registrant_name = "",
  starting_date = "",
  ending_date = "",
  tidy_result = TRUE,
  ignore_disclaimer = FALSE,
  min_amount = "",
  max_amount = ""
)
```

Arguments

`issues`	Optional, but when using multiple search terms, must be paired with iissue_joiner. Character vector of issues to search for (e.g., `c("tax", "energy", "OECD")`). Each term will be wrapped in double quotes and joined using `issue_joiner`.
`issue_joiner`	Optional, but must be used for multiple issues. Character string, either `"and"` or `"or"`, specifying whether to require all terms or any term in the search.
`year`	Optional. A character string specifying the year to search for results. – Only one year can be used.

`filing_period` Optional. A character string limiting results to a specific filing period. – Only the following can be used "first_quarter" "second_quarter" "third_quarter" "fourth_quarter". Very old filings sometimes can only be separated by filing period "mid_year" "year_end", but using those isn't recommended.

`client_name` Optional. A character string specifying the client an entity that filed its own disclosure, or the client a lobbyist discloses that it lobbied on behalf of.

`registrant_name` Optional. Filter results to filings for a specific registrant (lobbying firm or self-filer).

`starting_date` Optional. A character string specifying the start date for the search (Format is -YYYY-MM-DD).

`ending_date` Optional. A character string specifying the end date for the search (Format is -YYYY-MM-DD).

`tidy_result` Optional. A logical value that if `TRUE` (default), returns a reduced, tidy data frame with key columns. If `FALSE`, returns all available columns from api query.

`ignore_disclaimer` Optional. If `TRUE`, suppresses the printed disclaimer and guidance messages explaining the limitations of the data.

`min_amount` Optional. A character string specifying the minimum lobbying amount spent.

`max_amount` Optional. A character string specifying the maximum lobbying amount spent.

Details

This function queries the U.S. Senate Lobbying Disclosure API for filings that match the user-supplied criteria. Users can search by issue, client, registrant, filing period, and date range. The function automatically formats issue terms for the API and provides options to return a tidy or full data frame.

The returned data frame includes columns for registrant, client, filing type, income, expenses, year, date posted, document URL, and descriptions of lobbying activities by issue area. If `tidy_result = TRUE`, only a subset of columns is returned for easier analysis.

A disclaimer message is printed by default, reminding users to fact-check results and avoid double-counting. Set `ignore_disclaimer = TRUE` to suppress this message.

Value

A data frame of lobbying filings matching the query, with lobbying activities unnested and issues as columns. If `tidy_result = TRUE`, the data frame is reduced to key columns.

See Also

[set_senate_api_key()] for entering and storing the disclosures api key, [flag_dupes()] for handling duplicate filings or amendments filed in the same quarter that can cause doublecounting, and [flag_client_registrant_conflict()] for methods to prevent doublecounting when entities that file lobbying disclosures as registrants, but pay outside lobbying firms too, also show up as clients.

Examples

```
# Example function calls (require API key - see set_senate_api_key())

# Basic parameter validation (executable)
tryCatch({
  get_filings(issues = c("tax"), year = "invalid_year", ignore_disclaimer = TRUE)
}, error = function(e) {
  message("Expected error for invalid parameters")
})

## Not run:
# First, set your API key:
set_senate_api_key()

# Get raw third quarter filings that include any of the terms "tax", "trade", "health"
df <- get_filings(
  issues = c("tax", "trade", "health"),
  issue_joiner = "or",
  year = 2024,
  filing_period = "third_quarter",
  tidy_result = FALSE,
  ignore_disclaimer = FALSE
)

# Get a cleaner version of filings for a specific client
cleaner_df <- get_filings(
  issues = c("fees", "foods", "immigration"),
  issue_joiner = "and",
  client_name = "7 Eleven, Inc.",
  tidy_result = TRUE,
  ignore_disclaimer = FALSE
)

# Query filings for specific issues and client/registrant
df <- get_filings(
  issues = c("tax", "company", "bill"),
  issue_joiner = "or",
  filing_period = "first_quarter",
  client_name = "Chamber of Commerce of the U.S.A.",
  registrant_name = "Chamber of Commerce of the U.S.A.",
  ending_date = "2025-01-25",
  starting_date = "2021-04-01",
  min_amount = "",
  max_amount = "",
  tidy_result = TRUE,
  ignore_disclaimer = TRUE
)

## End(Not run)
```

`set_senate_api_key` *Set and Store Disclosure API Key*

Description

Prompts the user to enter their U.S. Senate Lobbying Disclosure API key and securely stores it using the `keyring` package. The stored key is used by [get_filings()] for authentication.

Usage

```
set_senate_api_key(api_key)
```

Arguments

`api_key`	Character. Optional. If provided, sets the API key directly. If omitted, the function prompts the user to enter the key.

Details

This function uses `keyring::key_set()` to securely store the API key under the name "senate_api_key". The function pauses briefly to allow the user to read instructions before prompting for input.

#' API keys can be requested at: https://lda.senate.gov/api/register/

Value

Invisibly returns `NULL`. The API key is stored securely in the system keyring.

See Also

[keyring::key_set()] for secure key storage, and [get_filings()] for using the stored API key.

Examples

```
# Simple validation test (executable)
if (is.function(set_senate_api_key)) {
  message("Function is available")
}

## Not run:
# Set or update the API key (prompts user for input)
set_senate_api_key()

# Set the API key directly (not recommended for interactive use)
set_senate_api_key("your_api_key_here")

## End(Not run)
```

Index