Package: tesouror (via r-universe)

October 29, 2026

Title Access Brazilian National Treasury Open Data APIs

Version 0.3.2

Description Provides a unified interface to access open data from the Brazilian National Treasury ('Tesouro Nacional') and related government APIs. Covers six data sources: 'SICONFI' <https://apidatalake.tesouro.gov.br/docs/siconfi/> for fiscal reports ('RREO', 'RGF', 'DCA', 'MSC') and entity information; 'CUSTOS' <https://apidatalake.tesouro.gov.br/docs/custos/> for federal government cost data; 'SADIPEM' <https://apidatalake.tesouro.gov.br/docs/sadipem/> for public debt and credit operations; 'Transferencias Constitucionais' <https://apiapex.tesouro.gov.br/aria/v1/transferencias_constitucionais/docs> for constitutional transfers to states and municipalities; 'SIORG' <https://estruturaorganizacional.dados.gov.br> for federal organizational structure; and 'SIOPE' ('FNDE'/'MEC') for education spending data. Features automatic pagination, in-memory caching, retry logic, and tidy output.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports cli (>= 3.6.0), dplyr (>= 1.1.0), httr2 (>= 1.0.0), janitor (>= 2.2.0), stringi (>= 1.7.0), stringr (>= 1.5.0), tibble (>= 3.2.0)

Suggests httptest2 (>= 1.0.0), knitr, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/StrategicProjects/tesouror, https://strategicprojects.github.io/tesouror/

BugReports https://github.com/StrategicProjects/tesouror/issues

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author André Leite [aut, cre] (ORCID: <https://orcid.org/0000-0002-4718-9766>), Marcos Wasiliew [aut] (ORCID: <https://orcid.org/0009-0004-4694-3159>), Hugo Vasconcelos [aut] (ORCID: <https://orcid.org/0000-0001-6249-0920>), Carlos Amorim [aut] (ORCID: <https://orcid.org/0000-0001-6315-8305>), Diogo Bezerra [aut] (ORCID: <https://orcid.org/0000-0002-1216-8674>), Tiago Pereira [aut], Fernando Barbalho [aut], Júlia Nascimento Barreto [aut] (ORCID: <https://orcid.org/0009-0004-2851-7770>)

Maintainer André Leite <leite@castlab.org>

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

Date/Publication 2026-10-29 21:15:33 UTC

RemoteUrl https://github.com/cran/tesouror

RemoteRef HEAD

RemoteSha 9daa275e4d627cdf173fd4444448c1581116c001

Contents

get_anexos . . . 3
get_custos_demais . . . 4
get_custos_depreciacao . . . 6
get_custos_pensionistas . . . 8
get_custos_pessoal_ativo . . . 10
get_custos_pessoal_inativo . . . 12
get_custos_transferencias . . . 14
get_dca_municipios . . . 16
get_dca_ufs . . . 17
get_entes . . . 19
get_extrato . . . 20
get_msc_controle . . . 21
get_msc_orcamentaria . . . 23
get_msc_patrimonial . . . 25
get_opc_cronograma_liberacoes . . . 26
get_opc_cronograma_pagamentos . . . 28
get_opc_taxa_cambio . . . 29
get_pvl . . . 30
get_pvl_tramitacao . . . 32
get_res_cdp . . . 33
get_res_cronograma_pagamentos . . . 34
get_rgf_municipios . . . 35
get_rgf_ufs . . . 37
get_rreo_municipios . . . 39
get_rreo_ufs . . . 41

get_siope_dados_gerais . . . 43
get_siope_despesas . . . 45
get_siope_despesas_funcao . . . 47
get_siope_indicadores . . . 49
get_siope_info_complementares . . . 51
get_siope_receitas . . . 53
get_siope_remuneracao . . . 54
get_siope_responsaveis . . . 56
get_siorg_estrutura . . . 58
get_siorg_orgaos . . . 59
get_siorg_unidade . . . 61
get_tc_estados . . . 62
get_tc_municipios . . . 63
get_tc_por_estados . . . 64
get_tc_por_estados_detalhe . . . 66
get_tc_por_municipio . . . 67
get_tc_por_municipio_detalhe . . . 69
get_tc_transferencias . . . 71
rreo_layout . . . 72
rreo_normalize_columns . . . 73
tesouror_clear_cache . . . 74
tidy_rreo . . . 74

Index **77**

`get_anexos` *Get report appendix reference table*

Description

Retrieves the reference table of report appendices (anexos) grouped by government sphere. This is a support table that describes which appendices are available for each report type (RREO, RGF, DCA, etc.).

Usage

```
get_anexos(use_cache = TRUE, verbose = FALSE, page_size = NULL, max_rows = Inf)

get_annexes(use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`page_size`	Integer or `NULL`. Number of rows per API page. If `NULL` (default), uses the API server default (5000 for SICONFI/SADIPEM).

`max_rows` Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).

Details

`get_annexes()` is an English alias for `get_anexos()`.

Value

A tibble with columns:

esfera Government sphere: `"U"` (Union), `"E"` (States), `"M"` (Municipalities).

demonstrativo Report type (e.g., `"RREO"`, `"RGF"`, `"DCA"`).

anexo Appendix name (e.g., `"RREO-Anexo 01"`).

See Also

Other SICONFI: `get_dca_municipios()`, `get_dca_ufs()`, `get_entes()`, `get_extrato()`, `get_msc_controle()`, `get_msc_orcamentaria()`, `get_msc_patrimonial()`, `get_rgf_municipios()`, `get_rgf_ufs()`, `get_rreo_municipios()`, `get_rreo_ufs()`

Examples

```
## Not run:
anexos <- get_anexos()

## End(Not run)
```

---

get_custos_demais *Get other cost data*

Description

Retrieves other (non-personnel) cost data for federal government organizations. All parameters are optional filters.

Usage

```
get_custos_demais(
  ano = NULL,
  mes = NULL,
  natureza_juridica = NULL,
  organizacao_n1 = NULL,
  organizacao_n2 = NULL,
  organizacao_n3 = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 500L,
```

```
  max_rows = Inf
)

get_costs_other(year = NULL, month = NULL,
  legal_nature = NULL, org_level1 = NULL, org_level2 = NULL,
  org_level3 = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = 500, max_rows = Inf)
```

Arguments

ano	Integer. Year of the record. Optional.
mes	Integer. Month of the record (1-12). Optional.
natureza_juridica	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional.
organizacao_n1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_orgaos() to look up codes. You can pass plain codes (e.g., 244) — they are automatically zero-padded to the 6-digit format the API expects ("000244"). Optional.
organizacao_n2	Character or integer. SIORG code for the second-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
organizacao_n3	Character or integer. SIORG code for the third-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to FALSE.
page_size	Integer. Number of rows per API page. Defaults to 500. The CUSTOS backend is slow on broad queries: smaller pages are more robust against HTTP 504 timeouts at the cost of a few extra round-trips. The server default of 250 is conservative; 1000+ may time out. If the package returns attr(result, "partial") = TRUE, lower this further or add a mes filter.
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
year	Integer. Year of the record. Optional. Maps to ano.
month	Integer. Month of the record (1-12). Optional. Maps to mes.
legal_nature	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional. Maps to natureza_juridica.
org_level1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_organizations() to look up codes. Plain codes (e.g., 244) are auto-padded to "000244". Optional. Maps to organizacao_n1.
org_level2	Character or integer. SIORG code (second level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n2.
org_level3	Character or integer. SIORG code (third level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n3.

Details

`get_costs_other()` is an English alias.

Value

A tibble with other cost data.

See Also

Other CUSTOS: `get_custos_depreciacao()`, `get_custos_pensionistas()`, `get_custos_pessoal_ativo()`, `get_custos_pessoal_inativo()`, `get_custos_transferencias()`

Examples

```
## Not run:
demais <- get_custos_demais(
  ano = 2023, mes = 6,
  organizacao_n1 = 244,  # MEC
  organizacao_n2 = 249   # INEP
)

## End(Not run)
```

---

get_custos_depreciacao

Get depreciation cost data

Description

Retrieves depreciation cost data for federal government organizations. All parameters are optional filters.

Usage

```
get_custos_depreciacao(
  ano = NULL,
  mes = NULL,
  natureza_juridica = NULL,
  organizacao_n1 = NULL,
  organizacao_n2 = NULL,
  organizacao_n3 = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 500L,
  max_rows = Inf
)

get_costs_depreciation(year = NULL, month = NULL,
```

```
  legal_nature = NULL, org_level1 = NULL, org_level2 = NULL,
  org_level3 = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = 500, max_rows = Inf)
```

Arguments

ano	Integer. Year of the record. Optional.
mes	Integer. Month of the record (1-12). Optional.
natureza_juridica	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional.
organizacao_n1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_orgaos() to look up codes. You can pass plain codes (e.g., 244) — they are automatically zero-padded to the 6-digit format the API expects ("000244"). Optional.
organizacao_n2	Character or integer. SIORG code for the second-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
organizacao_n3	Character or integer. SIORG code for the third-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to FALSE.
page_size	Integer. Number of rows per API page. Defaults to 500. The CUSTOS backend is slow on broad queries: smaller pages are more robust against HTTP 504 timeouts at the cost of a few extra round-trips. The server default of 250 is conservative; 1000+ may time out. If the package returns attr(result, "partial") = TRUE, lower this further or add a mes filter.
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
year	Integer. Year of the record. Optional. Maps to ano.
month	Integer. Month of the record (1-12). Optional. Maps to mes.
legal_nature	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional. Maps to natureza_juridica.
org_level1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_organizations() to look up codes. Plain codes (e.g., 244) are auto-padded to "000244". Optional. Maps to organizacao_n1.
org_level2	Character or integer. SIORG code (second level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n2.
org_level3	Character or integer. SIORG code (third level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n3.

Details

get_costs_depreciation() is an English alias.

Value

A tibble with depreciation cost data.

See Also

Other CUSTOS: `get_custos_demais()`, `get_custos_pensionistas()`, `get_custos_pessoal_ativo()`, `get_custos_pessoal_inativo()`, `get_custos_transferencias()`

Examples

```
## Not run:
deprec <- get_custos_depreciacao(
  ano = 2023, mes = 6,
  organizacao_n1 = 244,  # MEC
  organizacao_n2 = 249   # INEP
)

## End(Not run)
```

---

get_custos_pensionistas

Get pensioner cost data

Description

Retrieves cost data for federal government pensioners. All parameters are optional filters.

Usage

```
get_custos_pensionistas(
  ano = NULL,
  mes = NULL,
  natureza_juridica = NULL,
  organizacao_n1 = NULL,
  organizacao_n2 = NULL,
  organizacao_n3 = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 500L,
  max_rows = Inf
)

get_costs_pensioners(year = NULL, month = NULL,
  legal_nature = NULL, org_level1 = NULL, org_level2 = NULL,
  org_level3 = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = 500, max_rows = Inf)
```

Arguments

ano	Integer. Year of the record. Optional.
mes	Integer. Month of the record (1-12). Optional.
natureza_juridica	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional.
organizacao_n1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_orgaos() to look up codes. You can pass plain codes (e.g., 244) — they are automatically zero-padded to the 6-digit format the API expects ("000244"). Optional.
organizacao_n2	Character or integer. SIORG code for the second-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
organizacao_n3	Character or integer. SIORG code for the third-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to FALSE.
page_size	Integer. Number of rows per API page. Defaults to 500. The CUSTOS backend is slow on broad queries: smaller pages are more robust against HTTP 504 timeouts at the cost of a few extra round-trips. The server default of 250 is conservative; 1000+ may time out. If the package returns attr(result, "partial") = TRUE, lower this further or add a mes filter.
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
year	Integer. Year of the record. Optional. Maps to ano.
month	Integer. Month of the record (1-12). Optional. Maps to mes.
legal_nature	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional. Maps to natureza_juridica.
org_level1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_organizations() to look up codes. Plain codes (e.g., 244) are auto-padded to "000244". Optional. Maps to organizacao_n1.
org_level2	Character or integer. SIORG code (second level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n2.
org_level3	Character or integer. SIORG code (third level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n3.

Details

get_costs_pensioners() is an English alias.

Value

A tibble with pensioner cost data.

See Also

Other CUSTOS: get_custos_demais(), get_custos_depreciacao(), get_custos_pessoal_ativo(), get_custos_pessoal_inativo(), get_custos_transferencias()

Examples

```
## Not run:
pensionistas <- get_custos_pensionistas(
  ano = 2023, mes = 12,
  organizacao_n1 = 244,  # MEC
  organizacao_n2 = 249   # INEP
)

## End(Not run)
```

---

get_custos_pessoal_ativo

Get active staff cost data

Description

Retrieves cost data for active federal government staff. All parameters are optional filters.

Usage

```
get_custos_pessoal_ativo(
  ano = NULL,
  mes = NULL,
  natureza_juridica = NULL,
  organizacao_n1 = NULL,
  organizacao_n2 = NULL,
  organizacao_n3 = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 500L,
  max_rows = Inf
)

get_costs_active_staff(year = NULL, month = NULL,
  legal_nature = NULL, org_level1 = NULL, org_level2 = NULL,
  org_level3 = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = 500, max_rows = Inf)
```

Arguments

ano	Integer. Year of the record. Optional.
mes	Integer. Month of the record (1-12). Optional.

`natureza_juridica`
: Integer. Legal nature of the organization: `1` (Public Company), `2` (Public Foundation), `3` (Direct Administration), `4` (Autarchy), `6` (Mixed Economy Company). Optional.

`organizacao_n1`
: Character or integer. SIORG code for the top-level organization (Ministry level). Use `get_siorg_orgaos()` to look up codes. You can pass plain codes (e.g., `244`) — they are automatically zero-padded to the 6-digit format the API expects (`"000244"`). Optional.

`organizacao_n2`
: Character or integer. SIORG code for the second-level organization. Use `get_siorg_estrutura()` to browse sub-units. Automatically zero-padded. Optional.

`organizacao_n3`
: Character or integer. SIORG code for the third-level organization. Use `get_siorg_estrutura()` to browse sub-units. Automatically zero-padded. Optional.

`use_cache`
: Logical. If `TRUE` (default), uses an in-memory cache.

`verbose`
: Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `FALSE`.

`page_size`
: Integer. Number of rows per API page. Defaults to `500`. The CUSTOS backend is slow on broad queries: smaller pages are more robust against HTTP 504 timeouts at the cost of a few extra round-trips. The server default of 250 is conservative; 1000+ may time out. If the package returns `attr(result, "partial") = TRUE`, lower this further or add a `mes` filter.

`max_rows`
: Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).

`year`
: Integer. Year of the record. Optional. Maps to `ano`.

`month`
: Integer. Month of the record (1-12). Optional. Maps to `mes`.

`legal_nature`
: Integer. Legal nature of the organization: `1` (Public Company), `2` (Public Foundation), `3` (Direct Administration), `4` (Autarchy), `6` (Mixed Economy Company). Optional. Maps to `natureza_juridica`.

`org_level1`
: Character or integer. SIORG code for the top-level organization (Ministry level). Use `get_siorg_organizations()` to look up codes. Plain codes (e.g., `244`) are auto-padded to `"000244"`. Optional. Maps to `organizacao_n1`.

`org_level2`
: Character or integer. SIORG code (second level). Use `get_siorg_structure()` to browse sub-units. Auto-padded. Optional. Maps to `organizacao_n2`.

`org_level3`
: Character or integer. SIORG code (third level). Use `get_siorg_structure()` to browse sub-units. Auto-padded. Optional. Maps to `organizacao_n3`.

Details

Performance: The CUSTOS API is slow and unfiltered queries return hundreds of thousands of rows. Always filter by organization level (`organizacao_n1` + `organizacao_n2`) and/or `mes` to get manageable results. Use `max_rows` for quick tests.

`get_costs_active_staff()` is an English alias.

Value

A tibble with active staff cost data.

See Also

Other CUSTOS: get_custos_demais(), get_custos_depreciacao(), get_custos_pensionistas(), get_custos_pessoal_inativo(), get_custos_transferencias()

Examples

```
## Not run:
# WARNING: unfiltered queries return hundreds of thousands of rows
# and are very slow. Always filter by organization level and/or
# natureza_juridica + mes.

# Quick test: limit rows
ativos <- get_custos_pessoal_ativo(ano = 2023, mes = 6, max_rows = 100)

# Filter by ministry (N1) + entity (N2) for manageable results
# Use get_siorg_orgaos() to find SIORG codes
ativos_inep <- get_custos_pessoal_ativo(
  ano = 2023,
  organizacao_n1 = 244,  # MEC (auto-padded to "000244")
  organizacao_n2 = 249   # INEP (auto-padded to "000249")
)

## End(Not run)
```

---

get_custos_pessoal_inativo

Get retired staff cost data

Description

Retrieves cost data for retired (inactive) federal government staff. All parameters are optional filters.

Usage

```
get_custos_pessoal_inativo(
  ano = NULL,
  mes = NULL,
  natureza_juridica = NULL,
  organizacao_n1 = NULL,
  organizacao_n2 = NULL,
  organizacao_n3 = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 500L,
  max_rows = Inf
)

get_costs_retired_staff(year = NULL, month = NULL,
```

```
  legal_nature = NULL, org_level1 = NULL, org_level2 = NULL,
  org_level3 = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = 500, max_rows = Inf)
```

Arguments

ano	Integer. Year of the record. Optional.
mes	Integer. Month of the record (1-12). Optional.
natureza_juridica	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional.
organizacao_n1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_orgaos() to look up codes. You can pass plain codes (e.g., 244) — they are automatically zero-padded to the 6-digit format the API expects ("000244"). Optional.
organizacao_n2	Character or integer. SIORG code for the second-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
organizacao_n3	Character or integer. SIORG code for the third-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to FALSE.
page_size	Integer. Number of rows per API page. Defaults to 500. The CUSTOS backend is slow on broad queries: smaller pages are more robust against HTTP 504 timeouts at the cost of a few extra round-trips. The server default of 250 is conservative; 1000+ may time out. If the package returns attr(result, "partial") = TRUE, lower this further or add a mes filter.
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
year	Integer. Year of the record. Optional. Maps to ano.
month	Integer. Month of the record (1-12). Optional. Maps to mes.
legal_nature	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional. Maps to natureza_juridica.
org_level1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_organizations() to look up codes. Plain codes (e.g., 244) are auto-padded to "000244". Optional. Maps to organizacao_n1.
org_level2	Character or integer. SIORG code (second level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n2.
org_level3	Character or integer. SIORG code (third level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n3.

Details

get_costs_retired_staff() is an English alias.

Value

A tibble with retired staff cost data.

See Also

Other CUSTOS: `get_custos_demais()`, `get_custos_depreciacao()`, `get_custos_pensionistas()`, `get_custos_pessoal_ativo()`, `get_custos_transferencias()`

Examples

```
## Not run:
# Always filter to avoid slow, large queries
inativos <- get_custos_pessoal_inativo(
  ano = 2023, mes = 6,
  organizacao_n1 = 244,  # MEC
  organizacao_n2 = 249   # INEP
)

## End(Not run)
```

---

get_custos_transferencias

Get transfer cost data

Description

Retrieves transfer cost data for federal government organizations. All parameters are optional filters.

Usage

```
get_custos_transferencias(
  ano = NULL,
  mes = NULL,
  natureza_juridica = NULL,
  organizacao_n1 = NULL,
  organizacao_n2 = NULL,
  organizacao_n3 = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 500L,
  max_rows = Inf
)

get_costs_transfers(year = NULL, month = NULL,
  legal_nature = NULL, org_level1 = NULL, org_level2 = NULL,
  org_level3 = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = 500, max_rows = Inf)
```

Arguments

ano	Integer. Year of the record. Optional.
mes	Integer. Month of the record (1-12). Optional.
natureza_juridica	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional.
organizacao_n1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_orgaos() to look up codes. You can pass plain codes (e.g., 244) — they are automatically zero-padded to the 6-digit format the API expects ("000244"). Optional.
organizacao_n2	Character or integer. SIORG code for the second-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
organizacao_n3	Character or integer. SIORG code for the third-level organization. Use get_siorg_estrutura() to browse sub-units. Automatically zero-padded. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to FALSE.
page_size	Integer. Number of rows per API page. Defaults to 500. The CUSTOS backend is slow on broad queries: smaller pages are more robust against HTTP 504 timeouts at the cost of a few extra round-trips. The server default of 250 is conservative; 1000+ may time out. If the package returns attr(result, "partial") = TRUE, lower this further or add a mes filter.
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
year	Integer. Year of the record. Optional. Maps to ano.
month	Integer. Month of the record (1-12). Optional. Maps to mes.
legal_nature	Integer. Legal nature of the organization: 1 (Public Company), 2 (Public Foundation), 3 (Direct Administration), 4 (Autarchy), 6 (Mixed Economy Company). Optional. Maps to natureza_juridica.
org_level1	Character or integer. SIORG code for the top-level organization (Ministry level). Use get_siorg_organizations() to look up codes. Plain codes (e.g., 244) are auto-padded to "000244". Optional. Maps to organizacao_n1.
org_level2	Character or integer. SIORG code (second level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n2.
org_level3	Character or integer. SIORG code (third level). Use get_siorg_structure() to browse sub-units. Auto-padded. Optional. Maps to organizacao_n3.

Details

get_costs_transfers() is an English alias.

Value

A tibble with transfer cost data.

See Also

Other CUSTOS: get_custos_demais(), get_custos_depreciacao(), get_custos_pensionistas(), get_custos_pessoal_ativo(), get_custos_pessoal_inativo()

Examples

```
## Not run:
transf <- get_custos_transferencias(
  ano = 2023, mes = 6,
  organizacao_n1 = 244,  # MEC
  organizacao_n2 = 249   # INEP
)

## End(Not run)
```

---

get_dca_municipios *Get DCA data for all municipalities of a Brazilian state*

Description

Fetches annual accounts (DCA) for every municipality of `state_uf`, looping over `get_dca_ufs()` with fault tolerance. See `get_rreo_municipios()` for the rationale and behaviour of `on_error`.

Usage

```
get_dca_municipios(
  state_uf,
  an_exercicio,
  no_anexo = NULL,
  include_capital = TRUE,
  on_error = c("warn", "stop", "silent"),
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_annual_accounts_municipalities(state_uf, fiscal_year, appendix = NULL,
  include_capital = TRUE, on_error = c("warn", "stop", "silent"),
  use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`state_uf`	Character. Two-letter UF code (e.g., `"PE"`). **Required**.
`an_exercicio`	Integer. Fiscal year. **Required**.
`no_anexo`	Character. Appendix name filter (e.g., `"DCA-Anexo I-AB"`). Optional.

include_capital
: Logical. Include the state capital? Defaults to TRUE.

on_error
: Character. "warn" (default), "stop", or "silent".

use_cache
: Logical.

verbose
: Logical.

page_size
: Integer or NULL.

max_rows
: Numeric.

fiscal_year
: Integer. Fiscal year. **Required**. Maps to an_exercicio.

appendix
: Character. Appendix name filter. Optional. Maps to no_anexo.

Details

get_annual_accounts_municipalities() is an English-parameter alias.

Renamed in 0.3.0 from get_dca_for_state() for clarity. The old name still works but is deprecated.

Value

A tibble with all successful DCA rows. If any call failed, has an attribute "failed".

See Also

Other SICONFI: get_anexos(), get_dca_ufs(), get_entes(), get_extrato(), get_msc_controle(), get_msc_orcamentaria(), get_msc_patrimonial(), get_rgf_municipios(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
dca_pe <- get_dca_municipios(state_uf = "PE", an_exercicio = 2022)

## End(Not run)
```

---

get_dca_ufs *Get annual accounts data (DCA) for a single entity*

Description

Retrieves data from the Annual Accounts Declaration (DCA) or the legacy QDCC for a **single** entity (a state or a municipality, identified by id_ente) and fiscal year. To sweep every municipality of a state, use get_dca_municipios().

Usage

```
get_dca_ufs(
  an_exercicio,
  id_ente,
  no_anexo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_annual_accounts_ufs(fiscal_year, entity_id, appendix = NULL,
  use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

an_exercicio	Integer. Fiscal year (e.g., 2022). **Required**.
id_ente	Integer. IBGE code of the entity. **Required**.
no_anexo	Character. Appendix name filter (e.g., "DCA-Anexo I-AB"). Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouro.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
fiscal_year	Integer. Fiscal year (e.g., 2022). **Required**. Maps to an_exercicio.
entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.
appendix	Character. Appendix name filter (e.g., "DCA-Anexo I-AB"). Optional. Maps to no_anexo.

Details

get_annual_accounts_ufs() is an English-parameter alias for get_dca_ufs().

Renamed in 0.3.0 from get_dca() for clarity. The old name still works but is deprecated.

Value

A tibble with DCA/QDCC data including columns such as exercicio, instituicao, cod_ibge, uf, anexo, rotulo, coluna, cod_conta, conta, valor, and populacao.

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_entes(), get_extrato(), get_msc_controle(), get_msc_orcamentaria(), get_msc_patrimonial(), get_rgf_municipios(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
dca <- get_dca_ufs(an_exercicio = 2022, id_ente = 17)

## End(Not run)
```

`get_entes` *Get list of Brazilian government entities*

Description

Retrieves the complete list of government entities (entes) registered in the SICONFI system, including states, municipalities, and the Federal District.

Usage

```
get_entes(use_cache = TRUE, verbose = FALSE, page_size = NULL, max_rows = Inf)

get_entities(use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`page_size`	Integer or `NULL`. Number of rows per API page. If `NULL` (default), uses the API server default (5000 for SICONFI/SADIPEM).
`max_rows`	Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).

Details

`get_entities()` is an English alias for `get_entes()`.

Value

A tibble with columns:

cod_ibge IBGE code of the entity.

ente Name of the entity.

capital Whether the municipality is a state capital (1 = yes, 0 = no).

regiao Geographic region (`"SU"`, `"NE"`, `"NO"`, `"SE"`, `"CO"`, `"BR"`).

uf State abbreviation.

esfera Government sphere: `"M"`, `"E"`, `"U"`, `"D"`.

an_exercicio Year of the population data.

populacao Estimated population.

co_cnpj CNPJ of the entity.

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_dca_ufs(), get_extrato(), get_msc_controle(), get_msc_orcamentaria(), get_msc_patrimonial(), get_rgf_municipios(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
entes <- get_entes()

## End(Not run)
```

`get_extrato` *Get delivery status extract*

Description

Retrieves the extract of report deliveries for a given entity and reference year. Useful for checking which reports have been submitted and their status (approved, rectified, etc.).

Usage

```
get_extrato(
  id_ente,
  an_referencia,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_delivery_status(entity_id, year, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

id_ente	Integer. IBGE code of the entity. **Required**.
an_referencia	Integer. Reference year (e.g., 2022). **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.
year	Integer. Reference year (e.g., 2022). **Required**. Maps to an_referencia.

Details

`get_delivery_status()` is an English alias for `get_extrato()`.

Value

A tibble with delivery status data including columns such as `exercicio`, `cod_ibge`, `instituicao`, `entregavel`, `periodo`, `periodicidade`, `status_relatorio`, `data_status`, `forma_envio`, and `tipo_relatorio`.

See Also

Other SICONFI: `get_anexos()`, `get_dca_municipios()`, `get_dca_ufs()`, `get_entes()`, `get_msc_controle()`, `get_msc_orcamentaria()`, `get_msc_patrimonial()`, `get_rgf_municipios()`, `get_rgf_ufs()`, `get_rreo_municipios()`, `get_rreo_ufs()`

Examples

```
## Not run:
extrato <- get_extrato(id_ente = 17, an_referencia = 2022)

## End(Not run)
```

`get_msc_controle` *Get MSC control accounts data*

Description

Retrieves control accounts data (classes 7 and 8) from the Accounting Balances Matrix (MSC).

Usage

```
get_msc_controle(
  id_ente,
  an_referencia,
  me_referencia,
  co_tipo_matriz,
  classe_conta,
  id_tv,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_msc_control(entity_id, year, month, matrix_type, account_class,
  value_type, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

id_ente	Integer. IBGE code of the entity. **Required**.
an_referencia	Integer. Reference year. **Required**.
me_referencia	Integer. Reference month (1-12). **Required**.
co_tipo_matriz	Character. Matrix type: "MSCC" (monthly aggregate) or "MSCE" (annual closing). **Required**.
classe_conta	Integer. Account class: 7 or 8. **Required**.
id_tv	Character. Value type: "beginning_balance", "ending_balance", or "period_change". **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.
year	Integer. Reference year. **Required**. Maps to an_referencia.
month	Integer. Reference month (1-12). **Required**. Maps to me_referencia.
matrix_type	Character. Matrix type: "MSCC" (monthly aggregate) or "MSCE" (annual closing). **Required**. Maps to co_tipo_matriz.
account_class	Integer. Account class: 7 or 8. **Required**. Maps to classe_conta.
value_type	Character. Value type: "beginning_balance", "ending_balance", or "period_change". **Required**. Maps to id_tv.

Details

get_msc_control() is an English alias for get_msc_controle().

Value

A tibble with MSC control account data.

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_dca_ufs(), get_entes(), get_extrato(), get_msc_orcamentaria(), get_msc_patrimonial(), get_rgf_municipios(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
msc_ctrl <- get_msc_controle(
  id_ente = 17, an_referencia = 2022, me_referencia = 12,
  co_tipo_matriz = "MSCC", classe_conta = 8,
  id_tv = "ending_balance"
)

## End(Not run)
```

`get_msc_orcamentaria` *Get MSC budgetary accounts data*

Description

Retrieves budgetary accounts data (classes 5 and 6) from the Accounting Balances Matrix (MSC).

Usage

```
get_msc_orcamentaria(
  id_ente,
  an_referencia,
  me_referencia,
  co_tipo_matriz,
  classe_conta,
  id_tv,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_msc_budget(entity_id, year, month, matrix_type, account_class,
  value_type, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`id_ente`	Integer. IBGE code of the entity. **Required**.
`an_referencia`	Integer. Reference year. **Required**.
`me_referencia`	Integer. Reference month (1-12). **Required**.
`co_tipo_matriz`	Character. Matrix type: `"MSCC"` (monthly aggregate) or `"MSCE"` (annual closing). **Required**.
`classe_conta`	Integer. Account class: `5` or `6`. **Required**.
`id_tv`	Character. Value type: `"beginning_balance"`, `"ending_balance"`, or `"period_change"`. **Required**.

use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.
year	Integer. Reference year. **Required**. Maps to an_referencia.
month	Integer. Reference month (1-12). **Required**. Maps to me_referencia.
matrix_type	Character. Matrix type: "MSCC" (monthly aggregate) or "MSCE" (annual closing). **Required**. Maps to co_tipo_matriz.
account_class	Integer. Account class: 5 or 6. **Required**. Maps to classe_conta.
value_type	Character. Value type: "beginning_balance", "ending_balance", or "period_change". **Required**. Maps to id_tv.

Details

get_msc_budget() is an English alias for get_msc_orcamentaria().

Value

A tibble with MSC budgetary account data.

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_dca_ufs(), get_entes(), get_extrato(), get_msc_controle(), get_msc_patrimonial(), get_rgf_municipios(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
msc_orc <- get_msc_orcamentaria(
  id_ente = 17, an_referencia = 2022, me_referencia = 12,
  co_tipo_matriz = "MSCC", classe_conta = 6,
  id_tv = "period_change"
)

## End(Not run)
```

`get_msc_patrimonial` *Get MSC equity/asset accounts data*

Description

Retrieves equity and asset accounts data (classes 1 to 4) from the Accounting Balances Matrix (MSC).

Usage

```
get_msc_patrimonial(
  id_ente,
  an_referencia,
  me_referencia,
  co_tipo_matriz,
  classe_conta,
  id_tv,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_msc_equity(entity_id, year, month, matrix_type, account_class,
  value_type, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`id_ente`	Integer. IBGE code of the entity. **Required**.
`an_referencia`	Integer. Reference year. **Required**.
`me_referencia`	Integer. Reference month (1-12). **Required**.
`co_tipo_matriz`	Character. Matrix type: `"MSCC"` (monthly aggregate) or `"MSCE"` (annual closing). **Required**.
`classe_conta`	Integer. Account class: `1`, `2`, `3`, or `4`. **Required**.
`id_tv`	Character. Value type: `"beginning_balance"`, `"ending_balance"`, or `"period_change"`. **Required**.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`page_size`	Integer or `NULL`. Number of rows per API page. If `NULL` (default), uses the API server default (5000 for SICONFI/SADIPEM).
`max_rows`	Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).

entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.
year	Integer. Reference year. **Required**. Maps to an_referencia.
month	Integer. Reference month (1-12). **Required**. Maps to me_referencia.
matrix_type	Character. Matrix type: "MSCC" (monthly aggregate) or "MSCE" (annual closing). **Required**. Maps to co_tipo_matriz.
account_class	Integer. Account class: 1, 2, 3, or 4. **Required**. Maps to classe_conta.
value_type	Character. Value type: "beginning_balance", "ending_balance", or "period_change". **Required**. Maps to id_tv.

Details

get_msc_equity() is an English alias for get_msc_patrimonial().

Value

A tibble with MSC equity/asset account data.

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_dca_ufs(), get_entes(), get_extrato(), get_msc_controle(), get_msc_orcamentaria(), get_rgf_municipios(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
msc_pat <- get_msc_patrimonial(
  id_ente = 17, an_referencia = 2022, me_referencia = 12,
  co_tipo_matriz = "MSCC", classe_conta = 1,
  id_tv = "ending_balance"
)

## End(Not run)
```

---

get_opc_cronograma_liberacoes

Get credit operation release schedule

Description

Retrieves the release schedule for credit operations linked to a PVL request. The id_pleito can be obtained from get_pvl().

Usage

```
get_opc_cronograma_liberacoes(
  id_pleito,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_credit_release_schedule(request_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

id_pleito	Integer. Database ID from get_pvl(). **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
request_id	Integer. Database ID of the PVL request. Obtain from the id_pleito column of get_pvl() results. **Required**. Maps to id_pleito.

Details

get_credit_release_schedule() is an English alias.

Value

A tibble with release schedule data.

See Also

Other SADIPEM: get_opc_cronograma_pagamentos(), get_opc_taxa_cambio(), get_pvl(), get_pvl_tramitacao(), get_res_cdp(), get_res_cronograma_pagamentos()

Examples

```
## Not run:
pvl_pe <- get_pvl(uf = "PE")
sched <- get_opc_cronograma_liberacoes(id_pleito = pvl_pe$id_pleito[1])

## End(Not run)
```

`get_opc_cronograma_pagamentos` *Get credit operation payment schedule*

Description

Retrieves the payment schedule for credit operations linked to a PVL request. The `id_pleito` can be obtained from `get_pvl()`.

Usage

```
get_opc_cronograma_pagamentos(
  id_pleito,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_credit_payment_schedule(request_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`id_pleito`	Integer. Database ID from `get_pvl()`. **Required**.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`page_size`	Integer or `NULL`. Number of rows per API page. If `NULL` (default), uses the API server default (5000 for SICONFI/SADIPEM).
`max_rows`	Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).
`request_id`	Integer. Database ID of the PVL request. Obtain from the `id_pleito` column of `get_pvl()` results. **Required**. Maps to `id_pleito`.

Details

`get_credit_payment_schedule()` is an English alias.

Value

A tibble with payment schedule data.

See Also

Other SADIPEM: `get_opc_cronograma_liberacoes()`, `get_opc_taxa_cambio()`, `get_pvl()`, `get_pvl_tramitacao()`, `get_res_cdp()`, `get_res_cronograma_pagamentos()`

Examples

```
## Not run:
pvl_pe <- get_pvl(uf = "PE")
pgto <- get_opc_cronograma_pagamentos(id_pleito = pvl_pe$id_pleito[1])

## End(Not run)
```

`get_opc_taxa_cambio` *Get credit operation exchange rate data*

Description

Retrieves exchange rate data for credit operations linked to a PVL request. The `id_pleito` can be obtained from `get_pvl()`.

Usage

```
get_opc_taxa_cambio(
  id_pleito,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_credit_exchange_rate(request_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`id_pleito`	Integer. Database ID from `get_pvl()`. **Required**.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`page_size`	Integer or `NULL`. Number of rows per API page. If `NULL` (default), uses the API server default (5000 for SICONFI/SADIPEM).
`max_rows`	Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).
`request_id`	Integer. Database ID of the PVL request. Obtain from the `id_pleito` column of `get_pvl()` results. **Required**. Maps to `id_pleito`.

Details

`get_credit_exchange_rate()` is an English alias.

Value

A tibble with exchange rate data.

See Also

Other SADIPEM: `get_opc_cronograma_liberacoes()`, `get_opc_cronograma_pagamentos()`, `get_pvl()`, `get_pvl_tramitacao()`, `get_res_cdp()`, `get_res_cronograma_pagamentos()`

Examples

```
## Not run:
cambio <- get_opc_taxa_cambio(id_pleito = 40353)

## End(Not run)
```

`get_pvl` *Get public debt verification requests (PVL)*

Description

Retrieves data from the Public Debt Verification Letters (PVL) system. These are requests for approval of credit operations and debt by subnational entities. Use the resulting `id_pleito` column to query detail functions like `get_pvl_tramitacao()`, `get_opc_cronograma_liberacoes()`, etc.

Usage

```
get_pvl(
  uf = NULL,
  tipo_interessado = NULL,
  id_ente = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_debt_requests(state = NULL, entity_type = NULL,
  entity_id = NULL, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`uf`	Character. State abbreviation (e.g., `"PE"`). Optional.
`tipo_interessado`	Character. Type of requesting entity (e.g., `"Município"`, `"Estado"`). Optional.
`id_ente`	Integer. IBGE code of the requesting entity. Optional.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.

verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
state	Character. State abbreviation (e.g., "PE"). Optional. Maps to uf.
entity_type	Character. Type of requesting entity (e.g., "Município", "Estado"). Optional. Maps to tipo_interessado.
entity_id	Integer. IBGE code of the requesting entity. Optional. Maps to id_ente.

Details

get_debt_requests() is an English alias.

Value

A tibble with PVL data including id_pleito, tipo_interessado, interessado, cod_ibge, uf, status, tipo_operacao, finalidade, credor, valor, and more.

See Also

Other SADIPEM: get_opc_cronograma_liberacoes(), get_opc_cronograma_pagamentos(), get_opc_taxa_cambio(), get_pvl_tramitacao(), get_res_cdp(), get_res_cronograma_pagamentos()

Examples

```
## Not run:
# Search all PVLs for Pernambuco
pvl_pe <- get_pvl(uf = "PE")

# Search by IBGE code (Recife)
pvl_recife <- get_pvl(id_ente = 2611606)

# Use id_pleito from results for detail queries:
id <- pvl_pe$id_pleito[1]
pagamentos <- get_opc_cronograma_pagamentos(id_pleito = id)
cdp <- get_res_cdp(id_pleito = id)

# For PVL processing status, filter approved PVLs first:
deferidos <- pvl_pe[pvl_pe$status == "Deferido", ]
if (nrow(deferidos) > 0) {
  status <- get_pvl_tramitacao(id_pleito = deferidos$id_pleito[1])
}

## End(Not run)
```

`get_pvl_tramitacao` *Get PVL processing status (approved non-credit operations)*

Description

Retrieves processing status for PVL requests that were **approved (deferred)** and involve **non-credit operations**. This endpoint only returns data for PVLs with status `"Deferido"`. For most PVLs (e.g., archived, in progress, or credit operations), this will return an empty tibble.

Usage

```
get_pvl_tramitacao(
  id_pleito,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_pvl_status(request_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

`id_pleito`	Integer. Database ID of a **deferred** PVL request (from the `id_pleito` column of `get_pvl()`). **Required**.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`page_size`	Integer or `NULL`. Number of rows per API page. If `NULL` (default), uses the API server default (5000 for SICONFI/SADIPEM).
`max_rows`	Numeric. Maximum number of rows to return. Defaults to `Inf` (all rows). Useful for quick tests with large datasets (e.g., `max_rows = 100`).
`request_id`	Integer. Database ID of a **deferred** PVL request. Obtain from the `id_pleito` column of `get_pvl()` results, filtered by `status == "Deferido"`. **Required**. Maps to `id_pleito`.

Details

To find PVLs that have data in this endpoint, filter by `status == "Deferido"` in the results of `get_pvl()`.

`get_pvl_status()` is an English alias.

Value

A tibble with PVL processing status data. Returns an empty tibble if the PVL is not an approved non-credit operation.

See Also

Other SADIPEM: get_opc_cronograma_liberacoes(), get_opc_cronograma_pagamentos(), get_opc_taxa_cambio(), get_pvl(), get_res_cdp(), get_res_cronograma_pagamentos()

Examples

```
## Not run:
# Step 1: search PVLs and filter for approved ones
pvl_pe <- get_pvl(uf = "PE")
pvl_deferidos <- pvl_pe[pvl_pe$status == "Deferido", ]

# Step 2: pick an id_pleito from the APPROVED requests
id <- pvl_deferidos$id_pleito[1]

# Step 3: get processing status (only works for deferred PVLs)
status <- get_pvl_tramitacao(id_pleito = id)

## End(Not run)
```

---

get_res_cdp *Get debt capacity result (CDP)*

Description

Retrieves the Debt Capacity Result (Resultado da Capacidade de Pagamento) linked to a PVL request. The id_pleito can be obtained from get_pvl().

Usage

```
get_res_cdp(
  id_pleito,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_debt_capacity(request_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

id_pleito	Integer. Database ID from get_pvl(). **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouro.verbose", FALSE).

page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
request_id	Integer. Database ID of the PVL request. Obtain from the id_pleito column of get_pvl() results. **Required**. Maps to id_pleito.

Details

get_debt_capacity() is an English alias.

Value

A tibble with CDP result data.

See Also

Other SADIPEM: get_opc_cronograma_liberacoes(), get_opc_cronograma_pagamentos(), get_opc_taxa_cambio(), get_pvl(), get_pvl_tramitacao(), get_res_cronograma_pagamentos()

Examples

```
## Not run:
cdp <- get_res_cdp(id_pleito = 40353)

## End(Not run)
```

---

get_res_cronograma_pagamentos

Get debt payment schedule result

Description

Retrieves the payment schedule result linked to a PVL request. The id_pleito can be obtained from get_pvl().

Usage

```
get_res_cronograma_pagamentos(
  id_pleito,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_debt_payment_schedule(request_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

id_pleito	Integer. Database ID from get_pvl(). **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouro.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
request_id	Integer. Database ID of the PVL request. Obtain from the id_pleito column of get_pvl() results. **Required**. Maps to id_pleito.

Details

get_debt_payment_schedule() is an English alias.

Value

A tibble with payment schedule result data.

See Also

Other SADIPEM: get_opc_cronograma_liberacoes(), get_opc_cronograma_pagamentos(), get_opc_taxa_cambio(), get_pvl(), get_pvl_tramitacao(), get_res_cdp()

Examples

```
## Not run:
pgto_res <- get_res_cronograma_pagamentos(id_pleito = 40353)

## End(Not run)
```

---

get_rgf_municipios *Get RGF data for all municipalities of a Brazilian state*

Description

Fetches RGF data for every municipality of state_uf, looping over get_rgf_ufs() with fault tolerance. See get_rreo_municipios() for the rationale and behaviour of on_error.

Usage

```
get_rgf_municipios(
  state_uf,
  an_exercicio,
  in_periodicidade,
  nr_periodo,
  co_tipo_demonstrativo,
  no_anexo,
  co_poder,
  include_capital = TRUE,
  on_error = c("warn", "stop", "silent"),
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_fiscal_report_municipalities(state_uf, fiscal_year, periodicity,
  period, report_type, appendix, branch,
  include_capital = TRUE, on_error = c("warn", "stop", "silent"),
  use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

state_uf	Character. Two-letter UF code (e.g., "PE"). **Required**.
an_exercicio	Integer. Fiscal year. **Required**.
in_periodicidade	Character. "Q" (four-monthly) or "S" (semi-annual). **Required**.
nr_periodo	Integer. Period number. **Required**.
co_tipo_demonstrativo	Character. "RGF" or "RGF Simplificado". **Required**.
no_anexo	Character. Appendix name (e.g., "RGF-Anexo 01"). **Required**.
co_poder	Character. Government branch: "E", "L", "J", "M", "D". **Required**.
include_capital	Logical. Include the state capital? Defaults to TRUE.
on_error	Character. "warn" (default), "stop", or "silent".
use_cache	Logical.
verbose	Logical.
page_size	Integer or NULL.
max_rows	Numeric.
fiscal_year	Integer. Fiscal year. **Required**. Maps to an_exercicio.
periodicity	Character. "Q" or "S". **Required**. Maps to in_periodicidade.
period	Integer. Period number. **Required**. Maps to nr_periodo.

report_type	Character. "RGF" or "RGF Simplificado". **Required**. Maps to co_tipo_demonstrativo.
appendix	Character. Appendix name. **Required**. Maps to no_anexo.
branch	Character. Government branch. **Required**. Maps to co_poder.

Details

get_fiscal_report_municipalities() is an English-parameter alias.

Renamed in 0.3.0 from get_rgf_for_state() for clarity. The old name still works but is deprecated.

Value

A tibble with all successful RGF rows. If any call failed, has an attribute "failed".

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_dca_ufs(), get_entes(), get_extrato(), get_msc_controle(), get_msc_orcamentaria(), get_msc_patrimonial(), get_rgf_ufs(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
rgf_pe <- get_rgf_municipios(
  state_uf = "PE", an_exercicio = 2022,
  in_periodicidade = "Q", nr_periodo = 3,
  co_tipo_demonstrativo = "RGF", no_anexo = "RGF-Anexo 01",
  co_poder = "E"
)

## End(Not run)
```

---

get_rgf_ufs *Get Fiscal Management Report data (RGF) for a single entity*

Description

Retrieves data from the Fiscal Management Report (RGF) for a **single** entity (a state or a municipality, identified by id_ente). The RGF contains information about personnel expenses, debt, credit operations, and other fiscal indicators. To sweep every municipality of a state, use get_rgf_municipios().

Usage

```
get_rgf_ufs(
  an_exercicio,
  in_periodicidade,
  nr_periodo,
  co_tipo_demonstrativo,
  no_anexo,
  co_esfera,
  co_poder,
  id_ente,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_fiscal_report_ufs(fiscal_year, periodicity, period, report_type,
  appendix, sphere, branch, entity_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

an_exercicio	Integer. Fiscal year (e.g., 2022). **Required**.
in_periodicidade	Character. Periodicity: "Q" (four-monthly) or "S" (semi-annual). **Required**.
nr_periodo	Integer. Period number (1-3 for four-monthly, 1-2 for semi-annual). **Required**.
co_tipo_demonstrativo	Character. Report type: "RGF" or "RGF Simplificado". **Required**.
no_anexo	Character. Appendix name (e.g., "RGF-Anexo 01"). **Required**.
co_esfera	Character. Government sphere: "M" (municipalities), "E" (states), or "U" (union). **Required**.
co_poder	Character. Government branch: "E" (executive), "L" (legislative), "J" (judiciary), "M" (public ministry), "D" (public defender). **Required**.
id_ente	Integer. IBGE code of the entity. **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
fiscal_year	Integer. Fiscal year (e.g., 2022). **Required**. Maps to an_exercicio.
periodicity	Character. Periodicity: "Q" (quadrimester) or "S" (semester). **Required**. Maps to in_periodicidade.

period	Integer. Period number: 1-3 (quadrimester) or 1-2 (semester). **Required**. Maps to nr_periodo.
report_type	Character. Report type: "RGF" or "RGF Simplificado". **Required**. Maps to co_tipo_demonstrativo.
appendix	Character. Appendix name (e.g., "RGF-Anexo 01"). **Required**. Maps to no_anexo.
sphere	Character. Government sphere: "M", "E", or "U". **Required**. Maps to co_esfera.
branch	Character. Government branch: "E" (executive), "L" (legislative), "J" (judiciary), "M" (public ministry), "D" (public defender). **Required**. Maps to co_poder.
entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.

Details

get_fiscal_report_ufs() is an English-parameter alias for this function.

Renamed in 0.3.0 from get_rgf() for clarity. The old name still works but is deprecated.

Value

A tibble with RGF data.

See Also

Other SICONFI: get_anexos(), get_dca_municipios(), get_dca_ufs(), get_entes(), get_extrato(), get_msc_controle(), get_msc_orcamentaria(), get_msc_patrimonial(), get_rgf_municipios(), get_rreo_municipios(), get_rreo_ufs()

Examples

```
## Not run:
rgf <- get_rgf_ufs(
  an_exercicio = 2022, in_periodicidade = "Q", nr_periodo = 3,
  co_tipo_demonstrativo = "RGF", no_anexo = "RGF-Anexo 01",
  co_esfera = "E", co_poder = "E", id_ente = 17
)

## End(Not run)
```

---

get_rreo_municipios *Get RREO data for all municipalities of a Brazilian state*

Description

Fetches RREO data for every municipality of state_uf, looping over get_rreo_ufs() with fault tolerance: if an individual municipality call fails after all retries, the failure is recorded and the loop continues. Failed calls are returned in attr(result, "failed").

Usage

```
get_rreo_municipios(
  state_uf,
  an_exercicio,
  nr_periodo,
  co_tipo_demonstrativo,
  no_anexo,
  include_capital = TRUE,
  on_error = c("warn", "stop", "silent"),
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_budget_report_municipalities(state_uf, fiscal_year, period,
  report_type, appendix, include_capital = TRUE,
  on_error = c("warn", "stop", "silent"),
  use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

state_uf	Character. Two-letter UF code (e.g., "PE", "ES"). **Required**.
an_exercicio	Integer. Fiscal year. **Required**.
nr_periodo	Integer. Bimester (1-6). **Required**.
co_tipo_demonstrativo	Character. "RREO" or "RREO Simplificado". **Required**.
no_anexo	Character. Appendix name (e.g., "RREO-Anexo 01"). **Required**.
include_capital	Logical. Include the state capital? Defaults to TRUE.
on_error	Character. One of "warn" (default — log and continue), "stop" (abort on first failure), or "silent" (record but no message).
use_cache	Logical. If TRUE (default), uses the in-memory cache.
verbose	Logical. If TRUE, prints the full API URL for each call.
page_size	Integer or NULL. Rows per API page.
max_rows	Numeric. Maximum rows per municipality call.
fiscal_year	Integer. Fiscal year. **Required**. Maps to an_exercicio.
period	Integer. Bimester (1-6). **Required**. Maps to nr_periodo.
report_type	Character. "RREO" or "RREO Simplificado". **Required**. Maps to co_tipo_demonstrativo.
appendix	Character. Appendix name. **Required**. Maps to no_anexo.

Details

This is the recommended way to assemble a state-wide panel: it handles pagination per municipality, uses the cache, and surfaces partial failures instead of aborting on the first error (see SICONFI behaviour for entities that have not yet homologated a given report).

`get_budget_report_municipalities()` is an English-parameter alias.

Renamed in 0.3.0 from `get_rreo_for_state()` for clarity. The old name still works but is deprecated.

Value

A tibble with RREO rows for all successful municipalities. If any call failed, has an attribute `"failed"` (tibble with `iteration`, `id`, `error`).

See Also

Other SICONFI: `get_anexos()`, `get_dca_municipios()`, `get_dca_ufs()`, `get_entes()`, `get_extrato()`, `get_msc_controle()`, `get_msc_orcamentaria()`, `get_msc_patrimonial()`, `get_rgf_municipios()`, `get_rgf_ufs()`, `get_rreo_ufs()`

Examples

```
## Not run:
rreo_es <- get_rreo_municipios(
  state_uf = "ES", an_exercicio = 2021, nr_periodo = 6,
  co_tipo_demonstrativo = "RREO", no_anexo = "RREO-Anexo 01"
)
attr(rreo_es, "failed")

## End(Not run)
```

---

get_rreo_ufs — *Get Budget Execution Summary Report data (RREO) for a single entity*

Description

Retrieves data from the Budget Execution Summary Report (RREO) for a **single** entity (a state or a municipality, identified by `id_ente`). The RREO is published bimonthly and contains information about revenues, expenses, and other budgetary data. To sweep every municipality of a state, use `get_rreo_municipios()`.

Usage

```
get_rreo_ufs(
  an_exercicio,
  nr_periodo,
  co_tipo_demonstrativo,
```

```
  no_anexo,
  co_esfera = NULL,
  id_ente,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = NULL,
  max_rows = Inf
)

get_budget_report_ufs(fiscal_year, period, report_type, appendix,
  sphere = NULL, entity_id, use_cache = TRUE, verbose = FALSE,
  page_size = NULL, max_rows = Inf)
```

Arguments

an_exercicio	Integer. Fiscal year (e.g., 2022). **Required**.
nr_periodo	Integer. Bimester number (1-6). **Required**.
co_tipo_demonstrativo	Character. Report type: "RREO" or "RREO Simplificado". **Required**. Note: "RREO Simplificado" applies only to municipalities with fewer than 50,000 inhabitants that opted for simplified reporting.
no_anexo	Character. Appendix name (e.g., "RREO-Anexo 01"). **Required**.
co_esfera	Character or NULL. Government sphere: "M" (municipalities), "E" (states), or "U" (union). **Optional** — defaults to NULL, which omits the co_esfera filter from the request. Some entities (e.g. the Federal District constitutional fund) only return data when the sphere is *not* supplied, so leave this NULL if a sphere-filtered call comes back empty.
id_ente	Integer. IBGE code of the entity. **Required**.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouro.verbose", FALSE).
page_size	Integer or NULL. Number of rows per API page. If NULL (default), uses the API server default (5000 for SICONFI/SADIPEM).
max_rows	Numeric. Maximum number of rows to return. Defaults to Inf (all rows). Useful for quick tests with large datasets (e.g., max_rows = 100).
fiscal_year	Integer. Fiscal year (e.g., 2022). **Required**. Maps to an_exercicio.
period	Integer. Bimester number (1-6). **Required**. Maps to nr_periodo.
report_type	Character. Report type: "RREO" or "RREO Simplificado". **Required**. Maps to co_tipo_demonstrativo.
appendix	Character. Appendix name (e.g., "RREO-Anexo 01"). **Required**. Maps to no_anexo.
sphere	Character or NULL. Government sphere: "M" (municipalities), "E" (states), or "U" (union). **Optional** — defaults to NULL, which omits the sphere filter. Maps to co_esfera.
entity_id	Integer. IBGE code of the entity. **Required**. Maps to id_ente.

Details

`get_budget_report_ufs()` is an English-parameter alias for this function.

Renamed in 0.3.0 from `get_rreo()` for clarity. The old name still works but is deprecated.

Value

A tibble with RREO data including columns such as `exercicio`, `demonstrativo`, `periodo`, `periodicidade`, `instituicao`, `cod_ibge`, `uf`, `populacao`, `anexo`, `rotulo`, `coluna`, `cod_conta`, `conta`, and `valor`.

See Also

Other SICONFI: `get_anexos()`, `get_dca_municipios()`, `get_dca_ufs()`, `get_entes()`, `get_extrato()`, `get_msc_controle()`, `get_msc_orcamentaria()`, `get_msc_patrimonial()`, `get_rgf_municipios()`, `get_rgf_ufs()`, `get_rreo_municipios()`

Examples

```
## Not run:
rreo <- get_rreo_ufs(
  an_exercicio = 2022, nr_periodo = 6,
  co_tipo_demonstrativo = "RREO",
  no_anexo = "RREO-Anexo 01",
  co_esfera = "E", id_ente = 17
)

## End(Not run)
```

---

get_siope_dados_gerais

Get SIOPE general data

Description

Retrieves general data from SIOPE including demographics, GDP, revenues, expenses, and declaration metadata for municipalities and states.

Usage

```
get_siope_dados_gerais(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
```

```
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_general_data(year, period, state, type = NULL, use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

ano	Integer. Year of the data (e.g., 2023). **Required**.
periodo	Integer. Bimester period (1-6). **Required**.
uf	Character. State abbreviation (e.g., "PE"). **Required**.
tipo	Character. Convenience filter on the entity type, applied **server-side** via the DS_TIPO column so you don't download both the state and all its municipalities. Accepts "estado"/"uf" (state) or "municipio"/"municipios" (municipalities), accent- and case-insensitive. Combined with filter via and when both are given. Defaults to NULL (no type filter). type is the English alias.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL. Defaults to FALSE. Set globally with options(tesouror.verbose = TRUE).
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.
year	Integer. Year (e.g., 2023). **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation (e.g., "PE"). **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

Performance tip: use the `filter` parameter to narrow results on the server before downloading. For example, filtering by municipality name or IBGE code returns only the rows you need instead of all municipalities in the state.

`get_siope_general_data()` is an English alias.

Value

A tibble with 52 columns including `tipo`, `num_ano`, `sig_uf`, `cod_muni`, `nom_muni`, `num_popu`, revenue and expense values, PIB data, and declaration metadata.

See Also

Other SIOPE: `get_siope_despesas()`, `get_siope_despesas_funcao()`, `get_siope_indicadores()`, `get_siope_info_complementares()`, `get_siope_receitas()`, `get_siope_remuneracao()`, `get_siope_responsaveis(`

Examples

```
## Not run:
# General data for all municipalities in Pernambuco
dados <- get_siope_dados_gerais(ano = 2023, periodo = 6, uf = "PE")

# FAST: Filter to a single municipality (server-side)
recife <- get_siope_dados_gerais(
  ano = 2023, periodo = 6, uf = "PE",
  filter = "NOM_MUNI eq 'Recife'"
)

# Only the state-level row (skip the municipalities)
estado <- get_siope_dados_gerais(
  ano = 2023, periodo = 6, uf = "PE", tipo = "estado"
)

# Filter + select specific columns (use original API names!)
resumo <- get_siope_dados_gerais(
  ano = 2023, periodo = 6, uf = "PE",
  filter = "COD_MUNI eq 2611606",
  select = c("NOM_MUNI", "VAL_RECE_REAL", "VAL_DESP_PAGA")
)

## End(Not run)
```

`get_siope_despesas` *Get SIOPE education expenses data*

Description

Retrieves education spending data broken down by expense categories for municipalities and states.

Usage

```
get_siope_despesas(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_expenses(year, period, state, type = NULL, use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

ano	Integer. Year of the data (e.g., 2023). **Required**.
periodo	Integer. Bimester period (1-6). **Required**.
uf	Character. State abbreviation (e.g., "PE"). **Required**.
tipo	Character. Convenience filter on the entity type, applied **server-side** via the DS_TIPO column so you don't download both the state and all its municipalities. Accepts "estado"/"uf" (state) or "municipio"/"municipios" (municipalities), accent- and case-insensitive. Combined with filter via and when both are given. Defaults to NULL (no type filter). type is the English alias.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL. Defaults to FALSE. Set globally with options(tesouror.verbose = TRUE).
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.

year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_expenses() is an English alias.

Value

A tibble with 21 columns including expense categories and values.

See Also

Other SIOPE: get_siope_dados_gerais(), get_siope_despesas_funcao(), get_siope_indicadores(), get_siope_info_complementares(), get_siope_receitas(), get_siope_remuneracao(), get_siope_responsaveis(

Examples

```
## Not run:
desp <- get_siope_despesas(ano = 2023, periodo = 6, uf = "PE")

## End(Not run)
```

---

get_siope_despesas_funcao

Get SIOPE expenses by education function

Description

Retrieves education expenses broken down by government function (e.g., basic education, higher education).

Usage

```
get_siope_despesas_funcao(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
  filter = NULL,
```

```
  orderby = NULL,
  select = NULL
)

get_siope_expenses_by_function(year, period, state, type = NULL,
  use_cache = TRUE, verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

ano	Integer. Year of the data (e.g., 2023). **Required**.
periodo	Integer. Bimester period (1-6). **Required**.
uf	Character. State abbreviation (e.g., "PE"). **Required**.
tipo	Character. Convenience filter on the entity type, applied **server-side** via the DS_TIPO column so you don't download both the state and all its municipalities. Accepts "estado"/"uf" (state) or "municipio"/"municipios" (municipalities), accent- and case-insensitive. Combined with filter via and when both are given. Defaults to NULL (no type filter). type is the English alias.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL. Defaults to FALSE. Set globally with options(tesouror.verbose = TRUE).
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.
year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_expenses_by_function() is an English alias.

Value

A tibble with 12 columns including function codes and expense values.

See Also

Other SIOPE: `get_siope_dados_gerais()`, `get_siope_despesas()`, `get_siope_indicadores()`, `get_siope_info_complementares()`, `get_siope_receitas()`, `get_siope_remuneracao()`, `get_siope_responsaveis(`

Examples

```
## Not run:
desp_func <- get_siope_despesas_funcao(ano = 2023, periodo = 6, uf = "PE")

## End(Not run)
```

---

get_siope_indicadores *Get SIOPE education indicators*

Description

Retrieves education spending indicators such as percentage of revenue applied to education, FUNDEB compliance, etc.

Usage

```
get_siope_indicadores(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_indicators(year, period, state, type = NULL, use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

ano	Integer. Year of the data (e.g., 2023). **Required**.
periodo	Integer. Bimester period (1-6). **Required**.
uf	Character. State abbreviation (e.g., "PE"). **Required**.
tipo	Character. Convenience filter on the entity type, applied **server-side** via the DS_TIPO column so you don't download both the state and all its municipalities. Accepts "estado"/"uf" (state) or "municipio"/"municipios" (municipalities), accent- and case-insensitive. Combined with filter via and when both are given. Defaults to NULL (no type filter). type is the English alias.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL. Defaults to FALSE. Set globally with options(tesouror.verbose = TRUE).
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.
year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_indicators() is an English alias.

Value

A tibble with 13 columns including indicator codes and values.

See Also

Other SIOPE: get_siope_dados_gerais(), get_siope_despesas(), get_siope_despesas_funcao(), get_siope_info_complementares(), get_siope_receitas(), get_siope_remuneracao(), get_siope_responsaveis(

Examples

```
## Not run:
ind <- get_siope_indicadores(ano = 2023, periodo = 6, uf = "PE")

## End(Not run)
```

`get_siope_info_complementares` *Get SIOPE supplementary information*

Description

Retrieves supplementary information from SIOPE declarations.

Usage

```
get_siope_info_complementares(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_supplementary(year, period, state, type = NULL, use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

`ano`	Integer. Year of the data (e.g., `2023`). **Required**.
`periodo`	Integer. Bimester period (1-6). **Required**.
`uf`	Character. State abbreviation (e.g., `"PE"`). **Required**.
`tipo`	Character. Convenience filter on the entity type, applied **server-side** via the `DS_TIPO` column so you don't download both the state and all its municipalities. Accepts `"estado"`/`"uf"` (state) or `"municipio"`/`"municipios"` (municipalities), accent- and case-insensitive. Combined with `filter` via and when both are given. Defaults to `NULL` (no type filter). `type` is the English alias.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.

verbose	Logical. If TRUE, prints the full API URL. Defaults to FALSE. Set globally with options(tesouror.verbose = TRUE).
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.
year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_supplementary() is an English alias.

Value

A tibble with 10 columns.

See Also

Other SIOPE: get_siope_dados_gerais(), get_siope_despesas(), get_siope_despesas_funcao(), get_siope_indicadores(), get_siope_receitas(), get_siope_remuneracao(), get_siope_responsaveis()

Examples

```
## Not run:
info <- get_siope_info_complementares(ano = 2023, periodo = 6, uf = "PE")

## End(Not run)
```

`get_siope_receitas` *Get SIOPE education revenue data*

Description

Retrieves education revenue data including tax revenues, transfers, and FUNDEB contributions.

Usage

```
get_siope_receitas(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_revenues(year, period, state, type = NULL, use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

`ano`	Integer. Year of the data (e.g., `2023`). **Required**.
`periodo`	Integer. Bimester period (1-6). **Required**.
`uf`	Character. State abbreviation (e.g., `"PE"`). **Required**.
`tipo`	Character. Convenience filter on the entity type, applied **server-side** via the `DS_TIPO` column so you don't download both the state and all its municipalities. Accepts `"estado"`/`"uf"` (state) or `"municipio"`/`"municipios"` (municipalities), accent- and case-insensitive. Combined with `filter` via `and` when both are given. Defaults to `NULL` (no type filter). `type` is the English alias.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL. Defaults to `FALSE`. Set globally with `options(tesouror.verbose = TRUE)`.
`page_size`	Integer. Rows per page for OData pagination. Defaults to `1000`.
`max_rows`	Numeric. Maximum rows to return. Defaults to `Inf`.

filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.
year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_revenues() is an English alias.

Value

A tibble with 14 columns including revenue categories and values.

See Also

Other SIOPE: get_siope_dados_gerais(), get_siope_despesas(), get_siope_despesas_funcao(), get_siope_indicadores(), get_siope_info_complementares(), get_siope_remuneracao(), get_siope_responsaveis()

Examples

```
## Not run:
rec <- get_siope_receitas(ano = 2023, periodo = 6, uf = "PE")

## End(Not run)
```

---

get_siope_remuneracao *Get SIOPE staff compensation data*

Description

Retrieves compensation data for education professionals. This endpoint has an additional required parameter: mes / month for the specific month within the year.

Usage

```
get_siope_remuneracao(
  ano,
  periodo,
  mes,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_compensation(year, period, month, state, type = NULL,
  use_cache = TRUE, verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

ano	Integer. Year of the data (e.g., 2023). **Required**.
periodo	Integer. Bimester period (1-6). **Required**.
mes	Integer. Month of the fiscal year (1-12). **Required**.
uf	Character. State abbreviation (e.g., "PE"). **Required**.
tipo	Character. Convenience filter on the entity type, applied **server-side** via the DS_TIPO column. Accepts "estado"/"uf" or "municipio"/"municipios" (accent- and case-insensitive). Combined with filter via and. Defaults to NULL. type is the English alias.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL.
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.

year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
month	Integer. Month (1-12). **Required**. Maps to mes.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_compensation() is an English alias.

Value

A tibble with 19 columns including compensation categories and values for education professionals.

See Also

Other SIOPE: get_siope_dados_gerais(), get_siope_despesas(), get_siope_despesas_funcao(), get_siope_indicadores(), get_siope_info_complementares(), get_siope_receitas(), get_siope_responsaveis(

Examples

```
## Not run:
rem <- get_siope_remuneracao(ano = 2023, periodo = 6, mes = 12, uf = "PE")

## End(Not run)
```

---

get_siope_responsaveis

Get SIOPE officials/responsible persons data

Description

Retrieves data about the officials responsible for education spending declarations, including mayors, secretaries, and accountants.

Usage

```
get_siope_responsaveis(
  ano,
  periodo,
  uf,
  tipo = NULL,
  use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000L,
  max_rows = Inf,
```

```
  filter = NULL,
  orderby = NULL,
  select = NULL
)

get_siope_officials(year, period, state, type = NULL, use_cache = TRUE,
  verbose = FALSE,
  page_size = 1000, max_rows = Inf,
  filter = NULL, orderby = NULL, select = NULL)
```

Arguments

ano	Integer. Year of the data (e.g., 2023). **Required**.
periodo	Integer. Bimester period (1-6). **Required**.
uf	Character. State abbreviation (e.g., "PE"). **Required**.
tipo	Character. Convenience filter on the entity type, applied **server-side** via the DS_TIPO column so you don't download both the state and all its municipalities. Accepts "estado"/"uf" (state) or "municipio"/"municipios" (municipalities), accent- and case-insensitive. Combined with filter via and when both are given. Defaults to NULL (no type filter). type is the English alias.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL. Defaults to FALSE. Set globally with options(tesouror.verbose = TRUE).
page_size	Integer. Rows per page for OData pagination. Defaults to 1000.
max_rows	Numeric. Maximum rows to return. Defaults to Inf.
filter	Character. OData $filter expression to narrow results **on the server** (much faster than downloading everything). Uses OData syntax (e.g., "NOM_MUNI eq 'Recife'", "COD_MUNI eq 2611606"). Combine with and/or. Column names must use the **original API names** (uppercase), not the snake_case cleaned names. To discover valid names, run a max_rows = 1 query and use toupper(names(result)). Optional.
orderby	Character. OData $orderby expression to sort results (e.g., "NOM_MUNI asc", "NUM_POPU desc"). Uses original API column names (uppercase). Optional.
select	Character vector. Column names to return (reduces payload size). Uses original API column names (e.g., c("NOM_MUNI", "VAL_DECL")). If a column name is invalid the API returns HTTP 400. Optional.
year	Integer. Year. **Required**. Maps to ano.
period	Integer. Bimester (1-6). **Required**. Maps to periodo.
state	Character. State abbreviation. **Required**. Maps to uf.
type	Character. English alias for tipo: "state"/"uf" or "municipality"/"municipalities". Optional, defaults to NULL.

Details

get_siope_officials() is an English alias.

Value

A tibble with 35 columns including official names, CPFs, roles, and contact information.

See Also

Other SIOPE: get_siope_dados_gerais(), get_siope_despesas(), get_siope_despesas_funcao(), get_siope_indicadores(), get_siope_info_complementares(), get_siope_receitas(), get_siope_remuneracao()

Examples

```
## Not run:
resp <- get_siope_responsaveis(ano = 2023, periodo = 6, uf = "PE")

## End(Not run)
```

---

get_siorg_estrutura *Get organizational structure from SIORG*

Description

Retrieves the organizational structure tree for a specific SIORG unit, returning all sub-units (departments, secretariats, etc.) as a flat tibble.

Usage

```
get_siorg_estrutura(
  codigo_unidade,
  vinculados = NULL,
  use_cache = TRUE,
  verbose = FALSE
)

get_siorg_structure(unit_code, include_linked = NULL,
  use_cache = TRUE, verbose = FALSE)
```

Arguments

codigo_unidade	Integer or character. SIORG code of the unit. **Required**. Use get_siorg_orgaos() to find codes.
vinculados	Character. Return linked organs/entities? "SIM" or "NAO". Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL. Defaults to getOption("tesouror.verbose", FALSE).
unit_code	Integer or character. SIORG code. Maps to codigo_unidade.
include_linked	Character. "SIM" or "NAO". Maps to vinculados.

Details

Use the returned codes as `organizacao_n2` / `org_level2` and `organizacao_n3` / `org_level3` in CUSTOS API functions.

`get_siorg_structure()` is an English alias.

Value

A tibble with columns including `codigo_unidade`, `codigo_unidade_pai`, `nome`, `sigla`, `codigo_tipo_unidade`. Use `codigo_unidade_pai` to navigate the parent-child hierarchy.

See Also

Other SIORG: `get_siorg_orgaos()`, `get_siorg_unidade()`

Examples

```
## Not run:
# Get structure of AGU (code 46)
estrutura <- get_siorg_estrutura(codigo_unidade = 46)

## End(Not run)
```

`get_siorg_orgaos` *Get federal organizations from SIORG*

Description

Retrieves the list of federal government organizations (organs and entities) from the SIORG system. Use the returned SIORG codes as `organizacao_n1` / `org_level1` in CUSTOS API functions like `get_custos_pessoal_ativo()`.

Usage

```
get_siorg_orgaos(
  codigo_poder = NULL,
  codigo_esfera = NULL,
  use_cache = TRUE,
  verbose = FALSE
)

get_siorg_organizations(power_code = NULL, sphere_code = NULL,
  use_cache = TRUE, verbose = FALSE)
```

Arguments

codigo_poder	Integer. Power branch code: 1 = Executive, 2 = Legislative, 3 = Judiciary. Optional (default returns all).
codigo_esfera	Integer. Government sphere: 1 = Federal, 2 = State/District, 3 = Municipal. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Defaults to getOption("tesouror.verbose", FALSE).
power_code	Integer. Power branch: 1 = Executive, 2 = Legislative, 3 = Judiciary. Maps to codigo_poder.
sphere_code	Integer. Sphere: 1 = Federal. Maps to codigo_esfera.

Details

get_siorg_organizations() is an English alias.

Value

A tibble with columns including codigo_unidade (SIORG code — use as organizacao_n1 in CUSTOS), codigo_unidade_pai (parent code — for navigating hierarchy), nome, sigla, codigo_tipo_unidade ("orgao" or "entidade"), codigo_natureza_juridica, codigo_esfera, codigo_poder.

See Also

Other SIORG: get_siorg_estrutura(), get_siorg_unidade()

Examples

```
## Not run:
# List all federal executive branch organizations
orgaos <- get_siorg_orgaos(codigo_poder = 1, codigo_esfera = 1)

# Find a specific ministry by name
mec <- orgaos[grepl("Educação", orgaos$nome), ]
mec_code <- mec$codigo_unidade[mec$sigla == "MEC"]

# Use that code in CUSTOS queries (auto-padded to "000244")
custos <- get_custos_pessoal_ativo(
  ano = 2023, organizacao_n1 = mec_code
)

# Browse children: units whose parent is MEC
mec_filhos <- orgaos[orgaos$codigo_unidade_pai == mec_code, ]

## End(Not run)
```

`get_siorg_unidade` *Get details of a single SIORG unit*

Description

Retrieves summary data for a single organizational unit by its SIORG code. Returns a single-row tibble with the unit's details.

Usage

```
get_siorg_unidade(codigo_unidade, use_cache = TRUE, verbose = FALSE)

get_siorg_unit(unit_code, use_cache = TRUE, verbose = FALSE)
```

Arguments

`codigo_unidade`	Integer or character. SIORG code. **Required**.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL. Defaults to `getOption("tesouror.verbose", FALSE)`.
`unit_code`	Integer or character. SIORG code. Maps to `codigo_unidade`.

Details

`get_siorg_unit()` is an English alias.

Value

A tibble (single row) with unit details including `codigo_unidade`, `nome`, `sigla`, `codigo_tipo_unidade`, `codigo_unidade_pai`, `codigo_natureza_juridica`, and more.

See Also

Other SIORG: `get_siorg_estrutura()`, `get_siorg_orgaos()`

Examples

```
## Not run:
# Get details for AGU (code 46)
agu <- get_siorg_unidade(codigo_unidade = 46)

## End(Not run)
```

`get_tc_estados` *Get state dictionary*

Description

Retrieves the dictionary of states and their numeric codes. Use these codes in the `p_estado` / `state_code` parameter of other functions.

Usage

```
get_tc_estados(use_cache = TRUE, verbose = FALSE)

get_tc_states(use_cache = TRUE, verbose = FALSE)
```

Arguments

`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.

Details

These are **internal Treasury codes**, not IBGE codes.

`get_tc_states()` is an English alias.

Value

A tibble with columns `codigo` (numeric code) and `nome` (name) of each state.

See Also

Other Transferencias: `get_tc_municipios()`, `get_tc_por_estados()`, `get_tc_por_estados_detalhe()`, `get_tc_por_municipio()`, `get_tc_por_municipio_detalhe()`, `get_tc_transferencias()`

Other Transferencias dictionaries: `get_tc_municipios()`, `get_tc_transferencias()`

Examples

```
## Not run:
estados <- get_tc_estados()
# Use estados$codigo as values for p_estado in other functions

## End(Not run)
```

`get_tc_municipios` *Get municipality dictionary*

Description

Retrieves the dictionary of municipalities and their numeric codes. Use these codes in the `p_municipio` / `municipality` parameter of other functions.

Usage

```
get_tc_municipios(
  p_nome = NULL,
  p_uf = NULL,
  use_cache = TRUE,
  verbose = FALSE
)

get_tc_municipalities(name = NULL, state_code = NULL,
  use_cache = TRUE, verbose = FALSE)
```

Arguments

`p_nome`	Character. Partial municipality name for searching (e.g., `"Recife"`). Optional.
`p_uf`	Numeric or character. State code from `get_tc_estados()`. **Not** a state abbreviation or IBGE code. Optional.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`name`	Character. Partial municipality name for searching (e.g., `"Recife"`). Optional. Maps to `p_nome`.
`state_code`	Numeric or character. State code from `get_tc_estados()`. **Not** a state abbreviation or IBGE code. Optional. Maps to `p_uf`.

Details

These are **internal Treasury codes**, not IBGE codes. Filter by state using the numeric code from `get_tc_estados()`.

`get_tc_municipalities()` is an English alias.

Value

A tibble with municipality data including `codigo`, `codigo_uf`, and `nome`.

See Also

Other Transferencias: get_tc_estados(), get_tc_por_estados(), get_tc_por_estados_detalhe(), get_tc_por_municipio(), get_tc_por_municipio_detalhe(), get_tc_transferencias()

Other Transferencias dictionaries: get_tc_estados(), get_tc_transferencias()

Examples

```
## Not run:
# Step 1: find the state code (Treasury code, NOT IBGE)
estados <- get_tc_estados()
pe_code <- estados$codigo[estados$nome == "Pernambuco"]

# Step 2: list municipalities for that state
municipios <- get_tc_municipios(p_uf = pe_code)

# Or search by partial name
recife <- get_tc_municipios(p_nome = "Recife")

## End(Not run)
```

---

get_tc_por_estados *Get constitutional transfers by state*

Description

Retrieves constitutional transfer data aggregated by state.

Usage

```
get_tc_por_estados(
  p_estado = NULL,
  p_ano = NULL,
  p_mes = NULL,
  p_transferencia = NULL,
  p_sn_detalhar = NULL,
  use_cache = TRUE,
  verbose = FALSE
)

get_tc_by_state(state_code = NULL, year = NULL, month = NULL,
  transfer_type = NULL, detailed = NULL, use_cache = TRUE, verbose = FALSE)
```

Arguments

p_estado	State code(s) from get_tc_estados(). Accepts a vector or a colon-separated string. Optional.
p_ano	Year(s). Accepts a vector or colon-separated string. Optional.

p_mes	Month(s) (1-12). Accepts a vector or colon-separated string. Optional.
p_transferencia	Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional.
p_sn_detalhar	Character. Set to any value to include detailed breakdown. Omit for summary. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
state_code	State code(s) from get_tc_estados(). Accepts a vector (e.g., c(1, 2)) or colon-separated string ("1:2"). These are **Treasury codes**, not IBGE codes. Optional. Maps to p_estado.
year	Year(s). Accepts a vector or colon-separated string. Optional. Maps to p_ano.
month	Month(s) (1-12). Accepts a vector or colon-separated string. Optional. Maps to p_mes.
transfer_type	Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional. Maps to p_transferencia.
detailed	Character. Set to any value to include a detailed breakdown. Omit for summary. Optional. Maps to p_sn_detalhar.

Details

All codes are **internal Treasury codes** (not IBGE). Use the dictionary functions to look them up: get_tc_estados() for state codes and get_tc_transferencias() for transfer type codes.

Multi-value parameters accept either a colon-separated string ("1:2:3") or an R vector (c(1, 2, 3)).

get_tc_by_state() is an English alias.

Value

A tibble with transfer data by state.

See Also

Other Transferencias: get_tc_estados(), get_tc_municipios(), get_tc_por_estados_detalhe(), get_tc_por_municipio(), get_tc_por_municipio_detalhe(), get_tc_transferencias()

Examples

```
## Not run:
# Step 1: look up codes (Treasury codes, NOT IBGE)
estados <- get_tc_estados()
tipos   <- get_tc_transferencias()

pe_code <- estados$codigo[estados$nome == "Pernambuco"]

# Step 2: query (pass a vector or colon-separated string)
```

```
tc_pe <- get_tc_por_estados(p_estado = pe_code, p_ano = 2023)
tc_multi <- get_tc_por_estados(
  p_estado = c(1, 2), p_ano = 2023, p_mes = c(1, 2)
)

## End(Not run)
```

`get_tc_por_estados_detalhe` *Get detailed constitutional transfers by state*

Description

Retrieves detailed constitutional transfer data by state, with full breakdown of each transfer.

Usage

```
get_tc_por_estados_detalhe(
  p_estado = NULL,
  p_ano = NULL,
  p_mes = NULL,
  p_transferencia = NULL,
  use_cache = TRUE,
  verbose = FALSE
)

get_tc_by_state_detail(state_code = NULL, year = NULL,
  month = NULL, transfer_type = NULL, use_cache = TRUE, verbose = FALSE)
```

Arguments

`p_estado`	State code(s) from `get_tc_estados()`. Accepts a vector or colon-separated string. Optional.
`p_ano`	Year(s). Accepts a vector or colon-separated string. Optional.
`p_mes`	Month(s). Accepts a vector or colon-separated string. Optional.
`p_transferencia`	Transfer type code(s) from `get_tc_transferencias()`. Accepts a vector or colon-separated string. Optional.
`use_cache`	Logical. If `TRUE` (default), uses an in-memory cache.
`verbose`	Logical. If `TRUE`, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to `getOption("tesouror.verbose", FALSE)`.
`state_code`	State code(s) from `get_tc_estados()`. Accepts a vector or colon-separated string. **Treasury codes**, not IBGE. Optional. Maps to `p_estado`.
`year`	Year(s). Accepts a vector or colon-separated string. Optional. Maps to `p_ano`.
`month`	Month(s) (1-12). Accepts a vector or colon-separated string. Optional. Maps to `p_mes`.

transfer_type Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional. Maps to p_transferencia.

Details

All codes are **internal Treasury codes** (not IBGE). See get_tc_estados() and get_tc_transferencias() for dictionaries.

get_tc_by_state_detail() is an English alias.

Value

A tibble with detailed transfer data by state.

See Also

Other Transferencias: get_tc_estados(), get_tc_municipios(), get_tc_por_estados(), get_tc_por_municipio(), get_tc_por_municipio_detalhe(), get_tc_transferencias()

Examples

```
## Not run:
estados <- get_tc_estados()
pe_code <- estados$codigo[estados$nome == "Pernambuco"]
detalhe <- get_tc_por_estados_detalhe(p_estado = pe_code, p_ano = 2023)

## End(Not run)
```

---

get_tc_por_municipio *Get constitutional transfers by municipality*

Description

Retrieves constitutional transfer data for municipalities within states.

Usage

```
get_tc_por_municipio(
  p_estado = NULL,
  p_municipio = NULL,
  p_ano = NULL,
  p_mes = NULL,
  p_transferencia = NULL,
  p_sn_detalhar = NULL,
  page_size = 1000L,
  max_rows = Inf,
  use_cache = TRUE,
  verbose = FALSE
)
```

```
get_tc_by_municipality(state_code = NULL, municipality = NULL,
  year = NULL, month = NULL, transfer_type = NULL, detailed = NULL,
  page_size = 1000L, max_rows = Inf, use_cache = TRUE, verbose = FALSE)
```

Arguments

p_estado	State code(s) from get_tc_estados(). Accepts a vector or colon-separated string. Optional.
p_municipio	Municipality code(s) from get_tc_municipios(). Accepts a vector or colon-separated string. Optional.
p_ano	Year(s). Accepts a vector or colon-separated string. Optional.
p_mes	Month(s). Accepts a vector or colon-separated string. Optional.
p_transferencia	Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional.
p_sn_detalhar	Character. Set to "sim" (the documented value; "S" also works) to break every transfer into its components (for example FUNDEB - FPM, FUNDEB - ICMS). Optional. get_tc_por_municipio_detalhe() returns the same breakdown with the three ten-day installments.
page_size	Integer. Rows requested per page (the server pages this endpoint; 10 rows by default on the server side, 1000 here).
max_rows	Integer. Stop after this many rows. Default Inf.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
state_code	State code(s) from get_tc_estados(). Accepts a vector or colon-separated string. **Treasury codes**, not IBGE. Optional. Maps to p_estado.
municipality	Municipality code(s) from get_tc_municipios(). Accepts a vector or colon-separated string. **Treasury codes**. Optional. Maps to p_municipio.
year	Year(s). Accepts a vector or colon-separated string. Optional. Maps to p_ano.
month	Month(s) (1-12). Accepts a vector or colon-separated string. Optional. Maps to p_mes.
transfer_type	Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional. Maps to p_transferencia.
detailed	Character. Set to any value to include detail. Optional. Maps to p_sn_detalhar.

Details

All codes are **internal Treasury codes** (not IBGE). Use the dictionary functions to look them up: get_tc_estados() for state codes, get_tc_municipios() for municipality codes, and get_tc_transferencias() for transfer type codes.

Multi-value parameters accept either a colon-separated string ("1:2:3") or an R vector (c(1, 2, 3)).

`get_tc_by_municipality()` is an English alias.

The municipal endpoints are paginated: every page is fetched and the rows stacked. If a page after the first fails, the rows already fetched are returned with `attr(x, "partial") = TRUE` and `attr(x, "last_page_error")`.

Value

A tibble with transfer data by municipality (`uf`, `ano`, `mes`, `transferencia`, `codigo_siafi`, `co_ibge`, `municipio`, `valor`).

See Also

Other Transferencias: `get_tc_estados()`, `get_tc_municipios()`, `get_tc_por_estados()`, `get_tc_por_estados_detal`
`get_tc_por_municipio_detalhe()`, `get_tc_transferencias()`

Examples

```
## Not run:
# Step 1: look up codes (Treasury codes, NOT IBGE)
estados <- get_tc_estados()
pe_code <- estados$codigo[estados$nome == "Pernambuco"]
municipios <- get_tc_municipios(p_uf = pe_code)
recife_code <- municipios$codigo[municipios$nome == "Recife"]

# Step 2: query (pass vector or string)
tc <- get_tc_por_municipio(
  p_estado = pe_code,
  p_municipio = recife_code,
  p_ano = 2023
)

## End(Not run)
```

---

get_tc_por_municipio_detalhe

Get detailed constitutional transfers by municipality

Description

Retrieves detailed constitutional transfer data by municipality.

Usage

```
get_tc_por_municipio_detalhe(
  p_estado = NULL,
  p_municipio = NULL,
  p_ano = NULL,
  p_mes = NULL,
```

```
  p_transferencia = NULL,
  use_cache = TRUE,
  verbose = FALSE
)

get_tc_by_municipality_detail(state_code = NULL,
  municipality = NULL, year = NULL, month = NULL,
  transfer_type = NULL, use_cache = TRUE, verbose = FALSE)
```

Arguments

p_estado	State code(s) from get_tc_estados(). Accepts a vector or colon-separated string. Optional.
p_municipio	Municipality code(s) from get_tc_municipios(). Accepts a vector or colon-separated string. Optional.
p_ano	Year(s). Accepts a vector or colon-separated string. Optional.
p_mes	Month(s). Accepts a vector or colon-separated string. Optional.
p_transferencia	Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional.
use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouror.verbose", FALSE).
state_code	State code(s) from get_tc_estados(). Accepts a vector or colon-separated string. **Treasury codes**, not IBGE. Optional. Maps to p_estado.
municipality	Municipality code(s) from get_tc_municipios(). Accepts a vector or colon-separated string. **Treasury codes**. Optional. Maps to p_municipio.
year	Year(s). Accepts a vector or colon-separated string. Optional. Maps to p_ano.
month	Month(s) (1-12). Accepts a vector or colon-separated string. Optional. Maps to p_mes.
transfer_type	Transfer type code(s) from get_tc_transferencias(). Accepts a vector or colon-separated string. Optional. Maps to p_transferencia.

Details

All codes are **internal Treasury codes** (not IBGE). See get_tc_estados(), get_tc_municipios(), and get_tc_transferencias() for dictionaries.

get_tc_by_municipality_detail() is an English alias.

This endpoint is not paginated and returns one row per municipality, month and transfer component (sg_detalhe, e.g. FUNDEB/FPM), with the three ten-day installments (va_primeiro_dec, va_segundo_dec, va_terceiro_dec) and their total. A whole state for one month is about 2,900 rows and takes around 10 seconds.

Value

A tibble with detailed municipality transfer data (sg_uf, an_distribuicao, me_distribuicao, co_siafi, co_ibge, no_municipio, sg_detalhe, va_primeiro_dec, va_segundo_dec, va_terceiro_dec, total).

See Also

Other Transferencias: get_tc_estados(), get_tc_municipios(), get_tc_por_estados(), get_tc_por_estados_detal
get_tc_por_municipio(), get_tc_transferencias()

Examples

```
## Not run:
estados <- get_tc_estados()
pe_code <- estados$codigo[estados$nome == "Pernambuco"]
det <- get_tc_por_municipio_detalhe(p_estado = pe_code, p_ano = 2023)

## End(Not run)
```

---

get_tc_transferencias *Get transfer type dictionary*

Description

Retrieves the dictionary of constitutional transfer types and their numeric codes. Use these codes in the p_transferencia / transfer_type parameter of other functions.

Usage

```
get_tc_transferencias(use_cache = TRUE, verbose = FALSE)

get_tc_transfer_types(use_cache = TRUE, verbose = FALSE)
```

Arguments

use_cache	Logical. If TRUE (default), uses an in-memory cache.
verbose	Logical. If TRUE, prints the full API URL being called. Useful for debugging or testing in a browser. Defaults to getOption("tesouro.verbose", FALSE).

Details

These are **internal Treasury codes**, not IBGE codes.

get_tc_transfer_types() is an English alias.

Value

A tibble with columns codigo (code) and nome (name) of each transfer type.

See Also

Other Transferencias: get_tc_estados(), get_tc_municipios(), get_tc_por_estados(), get_tc_por_estados_detal
get_tc_por_municipio(), get_tc_por_municipio_detalhe()

Other Transferencias dictionaries: get_tc_estados(), get_tc_municipios()

Examples

```
## Not run:
tipos <- get_tc_transferencias()
# Use tipos$codigo as values for p_transferencia in other functions

## End(Not run)
```

`rreo_layout` *Return the bundled RREO layout reference table*

Description

RREO appendix names, account labels, and column suffixes drift across fiscal years (e.g., `RREO-Anexo 04.3 - RGPS` for 2019-2022 vs. `RREO-Anexo 04.4 - RGPS` for 2023+). The package ships a small reference table in `inst/extdata/rreo_layout.csv` that maps `(topic, regime, year_range)` to the correct appendix and account-matching key.

Usage

```
rreo_layout()
```

Details

Each rule matches either on a cleaned account label (`conta_match`) or on SICONFI's stable semantic account code (`cod_conta_match`, e.g. `"TotalReceitas"`); exactly one of the two is non-empty per row. Code matching is used where the free-text label is ambiguous — in Anexo 06, for instance, `"RESULTADO PRIMÁRIO (COM RPPS) - ..."` and `"RESULTADO PRIMÁRIO (SEM RPPS) - ..."` collapse to the same cleaned label, while their `cod_conta` values stay distinct.

tidy_rreo() uses this table to assemble layout-stable indicators across years; you can also use it directly to look up which appendix to fetch with get_rreo_ufs() for a given topic.

Value

A tibble with columns `topic`, `regime`, `first_year`, `last_year`, `co_esfera`, `no_anexo`, `conta_match`, `cod_conta_match`, `indicador`.

See Also

Other RREO tidy: rreo_normalize_columns(), tidy_rreo()

Examples

```
rreo_layout()
```

rreo_normalize_columns *Normalize the* coluna *field of a RREO tibble across years*

Description

SICONFI's RREO column labels drift over years: the same conceptual column appears as "DESPESAS LIQUIDADAS ATÉ O BIMESTRE / 2019" in 2019, "DESPESAS LIQUIDADAS ATÉ O BIMESTRE" (no year) in 2021-2022, and "DESPESAS LIQUIDADAS ATÉ O BIMESTRE / 2023" in 2023+. This helper adds two columns:

Usage

```
rreo_normalize_columns(data)
```

Arguments

data	A tibble returned by get_rreo_ufs() or get_rreo_municipios(). Must contain a coluna column.

Details

- coluna_padrao: the column label with any trailing "/ YYYY" or "EM YYYY" suffix removed (whitespace squished).
- coluna_ano: the year that appeared in the suffix (integer), or NA when no year was present. Useful for distinguishing the current-year column from a comparative previous-year column.

Value

The input tibble with coluna_padrao and coluna_ano appended.

See Also

Other RREO tidy: rreo_layout(), tidy_rreo()

Examples

```
demo <- tibble::tibble(
  coluna = c(
    "DESPESAS LIQUIDADAS ATÉ O BIMESTRE / 2023",
    "DESPESAS LIQUIDADAS ATÉ O BIMESTRE",
    "INSCRITAS EM RESTOS A PAGAR NÃO PROCESSADOS EM 2023"
  )
)
rreo_normalize_columns(demo)
```

tesouror_clear_cache *Clear the tesouror in-memory cache*

Description

Removes **all** cached API responses stored during the current R session. This applies to every API covered by the package: SICONFI, CUSTOS, SADIPEM, and Transferencias Constitucionais. All cached responses share the same in-memory store and are cleared together.

Usage

```
tesouror_clear_cache()
```

Value

Invisible NULL.

Examples

```
tesouror_clear_cache()
```

tidy_rreo *Tidy a RREO tibble by topic, reconciling layout drift across years*

Description

Filters a long RREO tibble (typically produced by get_rreo_ufs()) down to the rows that match a known indicator for topic (and optionally regime), using the rules in rreo_layout(). Account labels are matched on a year-stable, accent-folded key (Roman numerals and formula text are stripped before comparison), so the same call returns a coherent series across years even when SICONFI relabelled the appendix or account.

Usage

```
tidy_rreo(data, topic, regime = NULL)
```

Arguments

data	A tibble returned by get_rreo_ufs() or get_rreo_municipios(), with at least the columns exercicio, conta, coluna, valor.
topic	Character. Topic name (e.g., "previdencia"). See rreo_layout() for the supported set.
regime	Optional character. Filter to a subset of regimes within topic (e.g., "rgps", "rpps"). When NULL (default) all regimes are kept.

Details

Currently supported topics:

- "previdencia" — federal previdência (RGPS, RPPS civis, FCDF, militares inativos) for the União sphere. Anexos 04.1 / 04.2 / 04.3 / 04.4 of the RREO; the layout knows that the RGPS appendix moved from 04.3 (up to 2022) to 04.4 (2023+) and that civis/FCDF moved from 04.1 (up to 2022) to 04.2 (2023+).
- "rcl" — Receita Corrente Líquida (Anexo 03), all spheres. Regimes: "total" (2015+, União/states/municipalities) and, for states and municipalities from 2020 onwards, "ajustada_endividamento" and "ajustada_pessoal" (the adjusted RCL used for the debt and personnel spending limits; the União does not publish them, and the Roman numerals in their labels drifted in 2024 — the match key absorbs that).
- "balanco" — Balanço Orçamentário (Anexo 01), all spheres, 2015+. Matched on cod_conta, which stays stable across the 2020 relabeling ("TOTAL" became "TOTAL DAS RECEITAS" etc.) and disambiguates labels that repeat on both sides of the statement. Regimes: "receitas" and "despesas" (the subtotals excluding refinancing operations, the stable headline across eras), "receitas_correntes", "receitas_capital", "despesas_correntes", "despesas_capital" (main-budget categories; intra-budget rows carry their own codes and are not lumped in), "superavit" and "deficit" (each present only in the years the entity had one).
- "resultado" — Resultado Primário e Nominal (Anexo 06), all spheres. Matched on cod_conta: the 2023 state/municipal reform split the primary result into (COM RPPS) and (SEM RPPS) variants whose labels collapse to the same cleaned text. Regimes: "primario" (the headline above-the-line series: União 2015+, states/municipalities 2015-2022), "primario_com_rpps" / "primario_sem_rpps" (its state/municipal successors, 2023+), "primario_abaixo" (+ "primario_abaixo_sem_rpps"), "nominal" (2018+; before 2018 the nominal result lived in the old Anexo 05 as a derived debt variation and is not covered), "nominal_sem_rpps", "nominal_abaixo" (+ "nominal_abaixo_sem_rpps"), and "nominal_ajustado" (+ "nominal_ajustado_sem_rpps

Rules that match on cod_conta (see rreo_layout()) require the input to carry that column; the raw SICONFI output always does.

When the input has an anexo column (the raw SICONFI output always does), rows only match the layout rule for their own appendix. This avoids double counting with RREO-Anexo 14 (the simplified statement), which repeats the RCL accounts of Anexo 03 under the same labels — relevant when you fetched the whole RREO without no_anexo. Inputs without an anexo column skip this guard and match on account label and year alone.

Pull requests adding new topics to inst/extdata/rreo_layout.csv are welcome.

Value

A tibble with the matched rows, plus extra columns:

- indicador: stable indicator name (e.g., "resultado_previdenciario_rgps").
- regime: matched regime.
- coluna_padrao, coluna_ano: see rreo_normalize_columns().

See Also

Other RREO tidy: rreo_layout(), rreo_normalize_columns()

Examples

```
## Not run:
library(dplyr)
rreo <- purrr::map_dfr(2019:2023, \(yr) {
  get_rreo_ufs(an_exercicio = yr, nr_periodo = 6,
               co_tipo_demonstrativo = "RREO",
               no_anexo = rreo_layout()$no_anexo[
                 rreo_layout()$topic == "previdencia" &
                 rreo_layout()$regime == "rgps" &
                 yr >= rreo_layout()$first_year &
                 yr <= rreo_layout()$last_year
               ][1],
               co_esfera = "U", id_ente = 1)
})
tidy_rreo(rreo, topic = "previdencia", regime = "rgps") |>
  filter(coluna_padrao == "DESPESAS LIQUIDADAS ATÉ O BIMESTRE",
         is.na(coluna_ano) | coluna_ano == exercicio) |>
  select(exercicio, indicador, valor)

## End(Not run)
```

Index

∗ **CUSTOS**
  get_custos_demais, 4
  get_custos_depreciacao, 6
  get_custos_pensionistas, 8
  get_custos_pessoal_ativo, 10
  get_custos_pessoal_inativo, 12
  get_custos_transferencias, 14
∗ **RREO tidy**
  rreo_layout, 72
  rreo_normalize_columns, 73
  tidy_rreo, 74
∗ **SADIPEM**
  get_opc_cronograma_liberacoes, 26
  get_opc_cronograma_pagamentos, 28
  get_opc_taxa_cambio, 29
  get_pvl, 30
  get_pvl_tramitacao, 32
  get_res_cdp, 33
  get_res_cronograma_pagamentos, 34
∗ **SICONFI**
  get_anexos, 3
  get_dca_municipios, 16
  get_dca_ufs, 17
  get_entes, 19
  get_extrato, 20
  get_msc_controle, 21
  get_msc_orcamentaria, 23
  get_msc_patrimonial, 25
  get_rgf_municipios, 35
  get_rgf_ufs, 37
  get_rreo_municipios, 39
  get_rreo_ufs, 41
∗ **SIOPE**
  get_siope_dados_gerais, 43
  get_siope_despesas, 45
  get_siope_despesas_funcao, 47
  get_siope_indicadores, 49
  get_siope_info_complementares, 51
  get_siope_receitas, 53
  get_siope_remuneracao, 54
  get_siope_responsaveis, 56
∗ **SIORG dictionaries**
  get_siorg_orgaos, 59
∗ **SIORG**
  get_siorg_estrutura, 58
  get_siorg_orgaos, 59
  get_siorg_unidade, 61
∗ **Transferencias dictionaries**
  get_tc_estados, 62
  get_tc_municipios, 63
  get_tc_transferencias, 71
∗ **Transferencias**
  get_tc_estados, 62
  get_tc_municipios, 63
  get_tc_por_estados, 64
  get_tc_por_estados_detalhe, 66
  get_tc_por_municipio, 67
  get_tc_por_municipio_detalhe, 69
  get_tc_transferencias, 71

get_anexos, 3
get_anexos(), *17, 18*, *20–22*, *24*, *26*, *37*, *39*, *41*, *43*
get_annexes *(*get_anexos*)*, 3
get_annual_accounts_municipalities *(*get_dca_municipios*)*, 16
get_annual_accounts_ufs *(*get_dca_ufs*)*, 17
get_budget_report_municipalities *(*get_rreo_municipios*)*, 39
get_budget_report_ufs *(*get_rreo_ufs*)*, 41
get_costs_active_staff *(*get_custos_pessoal_ativo*)*, 10
get_costs_depreciation *(*get_custos_depreciacao*)*, 6
get_costs_other *(*get_custos_demais*)*, 4
get_costs_pensioners *(*get_custos_pensionistas*)*, 8

get_costs_retired_staff
(get_custos_pessoal_inativo), 12
get_costs_transfers
(get_custos_transferencias), 14
get_credit_exchange_rate
(get_opc_taxa_cambio), 29
get_credit_payment_schedule
(get_opc_cronograma_pagamentos), 28
get_credit_release_schedule
(get_opc_cronograma_liberacoes), 26
get_custos_demais, 4
get_custos_demais(), 8, 10, 12, 14, 16
get_custos_depreciacao, 6
get_custos_depreciacao(), 6, 10, 12, 14, 16
get_custos_pensionistas, 8
get_custos_pensionistas(), 6, 8, 12, 14, 16
get_custos_pessoal_ativo, 10
get_custos_pessoal_ativo(), 6, 8, 10, 14, 16, 59
get_custos_pessoal_inativo, 12
get_custos_pessoal_inativo(), 6, 8, 10, 12, 16
get_custos_transferencias, 14
get_custos_transferencias(), 6, 8, 10, 12, 14
get_dca_municipios, 16
get_dca_municipios(), 4, 17, 18, 20–22, 24, 26, 37, 39, 41, 43
get_dca_ufs, 17
get_dca_ufs(), 4, 16, 17, 20–22, 24, 26, 37, 39, 41, 43
get_debt_capacity (get_res_cdp), 33
get_debt_payment_schedule
(get_res_cronograma_pagamentos), 34
get_debt_requests (get_pvl), 30
get_delivery_status (get_extrato), 20
get_entes, 19
get_entes(), 4, 17, 18, 21, 22, 24, 26, 37, 39, 41, 43
get_entities (get_entes), 19
get_extrato, 20
get_extrato(), 4, 17, 18, 20, 22, 24, 26, 37, 39, 41, 43
get_fiscal_report_municipalities
(get_rgf_municipios), 35
get_fiscal_report_ufs (get_rgf_ufs), 37
get_msc_budget (get_msc_orcamentaria), 23
get_msc_control (get_msc_controle), 21
get_msc_controle, 21
get_msc_controle(), 4, 17, 18, 20, 21, 24, 26, 37, 39, 41, 43
get_msc_equity (get_msc_patrimonial), 25
get_msc_orcamentaria, 23
get_msc_orcamentaria(), 4, 17, 18, 20–22, 26, 37, 39, 41, 43
get_msc_patrimonial, 25
get_msc_patrimonial(), 4, 17, 18, 20–22, 24, 37, 39, 41, 43
get_opc_cronograma_liberacoes, 26
get_opc_cronograma_liberacoes(), 28, 30, 31, 33–35
get_opc_cronograma_pagamentos, 28
get_opc_cronograma_pagamentos(), 27, 30, 31, 33–35
get_opc_taxa_cambio, 29
get_opc_taxa_cambio(), 27, 28, 31, 33–35
get_pvl, 30
get_pvl(), 26–30, 32–35
get_pvl_status (get_pvl_tramitacao), 32
get_pvl_tramitacao, 32
get_pvl_tramitacao(), 27, 28, 30, 31, 34, 35
get_res_cdp, 33
get_res_cdp(), 27, 28, 30, 31, 33, 35
get_res_cronograma_pagamentos, 34
get_res_cronograma_pagamentos(), 27, 28, 30, 31, 33, 34
get_rgf_municipios, 35
get_rgf_municipios(), 4, 17, 18, 20–22, 24, 26, 37, 39, 41, 43
get_rgf_ufs, 37
get_rgf_ufs(), 4, 17, 18, 20–22, 24, 26, 35, 37, 41, 43
get_rreo_municipios, 39
get_rreo_municipios(), 4, 16–18, 20–22, 24, 26, 35, 37, 39, 41, 43, 73, 74
get_rreo_ufs, 41
get_rreo_ufs(), 4, 17, 18, 20–22, 24, 26, 37, 39, 41, 72–74
get_siope_compensation

(get_siope_remuneracao), 54
get_siope_dados_gerais, 43
get_siope_dados_gerais(), 47, 49, 50, 52, 54, 56, 58
get_siope_despesas, 45
get_siope_despesas(), 45, 49, 50, 52, 54, 56, 58
get_siope_despesas_funcao, 47
get_siope_despesas_funcao(), 45, 47, 50, 52, 54, 56, 58
get_siope_expenses (get_siope_despesas), 45
get_siope_expenses_by_function (get_siope_despesas_funcao), 47
get_siope_general_data (get_siope_dados_gerais), 43
get_siope_indicadores, 49
get_siope_indicadores(), 45, 47, 49, 52, 54, 56, 58
get_siope_indicators (get_siope_indicadores), 49
get_siope_info_complementares, 51
get_siope_info_complementares(), 45, 47, 49, 50, 54, 56, 58
get_siope_officials (get_siope_responsaveis), 56
get_siope_receitas, 53
get_siope_receitas(), 45, 47, 49, 50, 52, 56, 58
get_siope_remuneracao, 54
get_siope_remuneracao(), 45, 47, 49, 50, 52, 54, 58
get_siope_responsaveis, 56
get_siope_responsaveis(), 45, 47, 49, 50, 52, 54, 56
get_siope_revenues (get_siope_receitas), 53
get_siope_supplementary (get_siope_info_complementares), 51
get_siorg_estrutura, 58
get_siorg_estrutura(), 5, 7, 9, 11, 13, 15, 60, 61
get_siorg_organizations (get_siorg_orgaos), 59
get_siorg_organizations(), 5, 7, 9, 11, 13, 15
get_siorg_orgaos, 59
get_siorg_orgaos(), 5, 7, 9, 11, 13, 15, 58, 59, 61
get_siorg_structure (get_siorg_estrutura), 58
get_siorg_structure(), 5, 7, 9, 11, 13, 15
get_siorg_unidade, 61
get_siorg_unidade(), 59, 60
get_siorg_unit (get_siorg_unidade), 61
get_tc_by_municipality (get_tc_por_municipio), 67
get_tc_by_municipality_detail (get_tc_por_municipio_detalhe), 69
get_tc_by_state (get_tc_por_estados), 64
get_tc_by_state_detail (get_tc_por_estados_detalhe), 66
get_tc_estados, 62
get_tc_estados(), 63–72
get_tc_municipalities (get_tc_municipios), 63
get_tc_municipios, 63
get_tc_municipios(), 62, 65, 67–72
get_tc_por_estados, 64
get_tc_por_estados(), 62, 64, 67, 69, 71, 72
get_tc_por_estados_detalhe, 66
get_tc_por_estados_detalhe(), 62, 64, 65, 69, 71, 72
get_tc_por_municipio, 67
get_tc_por_municipio(), 62, 64, 65, 67, 71, 72
get_tc_por_municipio_detalhe, 69
get_tc_por_municipio_detalhe(), 62, 64, 65, 67–69, 72
get_tc_states (get_tc_estados), 62
get_tc_transfer_types (get_tc_transferencias), 71
get_tc_transferencias, 71
get_tc_transferencias(), 62, 64–71

rreo_layout, 72
rreo_layout(), 73–75
rreo_normalize_columns, 73
rreo_normalize_columns(), 72, 75

tesouror_clear_cache, 74
tibble, 4, 6, 8, 9, 11, 14, 15, 17–19, 21, 22, 24, 26–28, 30–32, 34, 35, 37, 39, 41,

43, *45*, *47*, *49*, *50*, *52*, *54*, *56*, *58–63*, *65*, *67*, *69*, *71*, *72*, *75*
tidy_rreo, 74
tidy_rreo(), *72*, *73*