

# Package ‘eurlex’

May 25, 2022

**Type** Package

**Title** Retrieve Data on European Union Law

**Version** 0.4.2

**Description** Access to data on European Union laws and court decisions made easy with pre-defined 'SPARQL' queries and 'GET' requests.

**License** GPL-3

**Encoding** UTF-8

**Depends** R (>= 3.4.0)

**Imports** magrittr, dplyr, xml2, tidyr, httr, curl, rvest, rlang, stringr, readr, pdftools, antiword

**RoxygenNote** 7.1.2

**Suggests** knitr, rmarkdown, tidytext, wordcloud, purrr, ggplot2, testthat (>= 3.0.0)

**URL** <https://michalovadek.github.io/eurlex/>

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Michal Ovadek [aut, cre, cph] (<<https://orcid.org/0000-0002-2552-2580>>)

**Maintainer** Michal Ovadek <michal.ovadek@gmail.com>

**Repository** CRAN

**Date/Publication** 2022-05-25 10:20:02 UTC

## R topics documented:

elx_council_votes . . . . .	2
elx_curia_list . . . . .	2
elx_download_xml . . . . .	3
elx_fetch_data . . . . .	4
elx_label_eurovoc . . . . .	5
elx_make_query . . . . .	6
elx_run_query . . . . .	8

**Index****9**


---

elx_council_votes	<i>Retrieve Council votes on EU acts</i>
-------------------	--

---

**Description**

Executes a SPARQL query to the Council's endpoint.

**Usage**

```
elx_council_votes()
```

**Value**

A data frame with Council votes on EU acts.

**Examples**

```
votes <- elx_council_votes()
```

---

elx_curia_list	<i>Scrape list of court cases from Curia</i>
----------------	--

---

**Description**

Harvests data from lists of EU court cases from curia.europa.eu. CELEX identifiers are extracted from hyperlinks where available.

**Usage**

```
elx_curia_list(
  data = c("all", "ecj_old", "ecj_new", "gc_all", "cst_all"),
  parse = TRUE
)
```

**Arguments**

data	Data to be scraped from four separate lists of cases maintained by Curia, defaults to "all" which contains cases from Court of Justice, General Court and Civil Service Tribunal.
parse	If 'TRUE', references to cases and appeals are parsed out from 'case_info' into separate columns

**Value**

A data frame containing case identifiers and information as character columns. Where the case id contains a hyperlink to Eur-Lex, the CELEX identifier is retrieved as well. Hyperlinks to Eur-Lex disappeared from more recent cases.

**Examples**

```
elx_curia_list(data = "cst_all")
```

---

elx_download_xml	<i>Download XML notice associated with a URL</i>
------------------	--

---

**Description**

Downloads an XML notice of a given type associated with a Cellar resource.

**Usage**

```
elx_download_xml(  
  url,  
  file = paste(basename(url), ".xml", sep = ""),  
  notice = c("tree", "branch", "object"),  
  language_1 = "en",  
  language_2 = "fr",  
  language_3 = "de",  
  mode = "wb"  
)
```

**Arguments**

url	A valid url as character vector of length one based on a resource identifier such as CELEX or Cellar URI.
file	A character string with the name where the downloaded file is saved.
notice	The type of notice requested controls what kind of metadata are returned.
language_1	The priority language in which the data will be attempted to be retrieved, in ISO 639 2-char code
language_2	If data not available in 'language_1', try 'language_2'
language_3	If data not available in 'language_2', try 'language_3'
mode	A character string specifying the mode with which to write the file. Useful values are "w", "wb" (binary), "a" (append) and "ab".

**Details**

To retrieve all identifiers associated with a url, use `elx_fetch_data(type = "ids")`.

**Value**

Path of downloaded file (invisibly) if server validates request (http status code has to be 200). For more information about notices, see Cellar documentation.

**Examples**

```
elx_download_xml(url = "http://publications.europa.eu/resource/celex/32014R0001", notice = "object")
```

---

elx_fetch_data	<i>Retrieve additional data on EU documents</i>
----------------	---

---

**Description**

Wraps http::GET with pre-specified headers and parses retrieved data.

**Usage**

```
elx_fetch_data(
  url,
  type = c("title", "text", "ids", "notice"),
  notice = c("tree", "branch", "object"),
  language_1 = "en",
  language_2 = "fr",
  language_3 = "de",
  include_breaks = TRUE
)
```

**Arguments**

url	A valid url as character vector of length one based on a resource identifier such as CELEX or Cellar URI.
type	The type of data to be retrieved. When type = "text", the returned list contains named elements reflecting the source of each text. When type = "notice", the results return an XML notice associated with the url.
notice	If type = "notice", controls what kind of metadata are returned by the notice.
language_1	The priority language in which the data will be attempted to be retrieved, in ISO 639 2-char code
language_2	If data not available in 'language_1', try 'language_2'
language_3	If data not available in 'language_2', try 'language_3'
include_breaks	If TRUE, text includes tags showing where pages ("—pagebreak—", for pdfs) and documents ("—documentbreak—") were concatenated

**Value**

A character vector of length one containing the result. When 'type = "text"', named character vector where the name contains the source of the text.

**Examples**

```
elx_fetch_data(url = "http://publications.europa.eu/resource/celex/32014R0001", type = "title")
```

---

elx_label_eurovoc	<i>Label EuroVoc concepts</i>
-------------------	-------------------------------

---

**Description**

Create a look-up table with labels for EuroVoc concept URIs. Only unique identifiers are returned.

**Usage**

```
elx_label_eurovoc(uri_eurovoc = "", alt_labels = FALSE, language = "en")
```

**Arguments**

uri_eurovoc	Character vector with valid EuroVoc URIs
alt_labels	If 'TRUE', results include comma-separated alternative labels in addition to the preferred label
language	Language in which to return the labels, in ISO 639 2-char code

**Value**

A 'tibble' containing EuroVoc unique concept identifiers and labels.

**Examples**

```
elx_label_eurovoc(uri_eurovoc = "http://eurovoc.europa.eu/5760", alt_labels = TRUE, language = "fr")
elx_label_eurovoc(uri_eurovoc = c("http://eurovoc.europa.eu/5760", "http://eurovoc.europa.eu/576"))
```

---

elx_make_query	<i>Create SPARQL queries</i>
----------------	------------------------------

---

### Description

Generates pre-defined or manual SPARQL queries to retrieve document ids from Cellar. List of available resource types: <http://publications.europa.eu/resource/authority/resource-type> . Note that not all resource types are compatible with default parameter values.

### Usage

```
elx_make_query(
  resource_type = c("any", "directive", "regulation", "decision", "recommendation",
    "intagr", "caselaw", "manual", "proposal", "national_impl"),
  manual_type = "",
  directory = NULL,
  sector = NULL,
  include_corrigenda = FALSE,
  include_celex = TRUE,
  include_lbs = FALSE,
  include_date = FALSE,
  include_date_force = FALSE,
  include_date_endvalid = FALSE,
  include_date_transpos = FALSE,
  include_date_lodged = FALSE,
  include_force = FALSE,
  include_eurovoc = FALSE,
  include_author = FALSE,
  include_citations = FALSE,
  include_court_procedure = FALSE,
  include_directory = FALSE,
  include_sector = FALSE,
  include_ecli = FALSE,
  include_judge_rapporteur = FALSE,
  include_advocate_general = FALSE,
  include_court_formation = FALSE,
  include_court_scholarship = FALSE,
  include_proposal = FALSE,
  order = FALSE,
  limit = NULL
)
```

### Arguments

resource_type	Type of resource to be retrieved via SPARQL query
manual_type	Define manually the type of resource to be retrieved

directory	Restrict the results to a given directory code
sector	Restrict the results to a given sector code
include_corrigena	If 'TRUE', results include corrigenda
include_celex	If 'TRUE', results include CELEX identifier for each resource URI
include_lbs	If 'TRUE', results include legal bases of legislation
include_date	If 'TRUE', results include document date
include_date_force	If 'TRUE', results include date of entry into force
include_date_endvalid	If 'TRUE', results include date of end of validity
include_date_transpos	If 'TRUE', results include date of transposition deadline for directives
include_date_lodged	If 'TRUE', results include date a court case was lodged with the court
include_force	If 'TRUE', results include whether legislation is in force
include_eurovoc	If 'TRUE', results include EuroVoc descriptors of subject matter
include_author	If 'TRUE', results include document author(s)
include_citations	If 'TRUE', results include citations (CELEX-labelled)
include_court_procedure	If 'TRUE', results include type of court procedure and outcome
include_directory	If 'TRUE', results include the Eur-Lex directory code
include_sector	If 'TRUE', results include the Eur-Lex sector code
include_ecli	If 'TRUE', results include the ECLI identifier for court documents
include_judge_rapporteur	If 'TRUE', results include the Judge-Rapporteur
include_advocate_general	If 'TRUE', results include the Advocate General
include_court_formation	If 'TRUE', results include the court formation
include_court_scholarship	If 'TRUE', results include court-curated relevant scholarship
include_proposal	If 'TRUE', results include the CELEX of the proposal of the adopted legal act
order	Order results by ids
limit	Limit the number of results, for testing purposes mainly

**Value**

A character string containing the SPARQL query

**Examples**

```

elx_make_query(resource_type = "directive", include_date = TRUE, include_force = TRUE)
elx_make_query(resource_type = "regulation", include_corrigena = TRUE, order = TRUE)
elx_make_query(resource_type = "caselaw")
elx_make_query(resource_type = "manual", manual_type = "SWD")

```

---

elx_run_query	<i>Execute SPARQL queries</i>
---------------	-------------------------------

---

**Description**

Executes cURL request to a pre-defined endpoint of the EU Publications Office. Relies on `elx_make_query` to generate valid SPARQL queries. Results are capped at 1 million rows.

**Usage**

```

elx_run_query(
  query = "",
  endpoint = "http://publications.europa.eu/webapi/rdf/sparql"
)

```

**Arguments**

query	A valid SPARQL query specified by <code>'elx_make_query()'</code> or manually
endpoint	SPARQL endpoint

**Value**

A data frame containing the results of the SPARQL query. Column `'work'` contains the Cellar URI of the resource.

**Examples**

```

elx_run_query(elx_make_query("directive", include_force = TRUE))

```



# Index

[elx\\_council\\_votes](#), 2  
[elx\\_curia\\_list](#), 2  
[elx\\_download\\_xml](#), 3  
[elx\\_fetch\\_data](#), 4  
[elx\\_label\\_eurovoc](#), 5  
[elx\\_make\\_query](#), 6  
[elx\\_run\\_query](#), 8