Package 'rconvertu'

May 17, 2026

Title From/to Classification Converter

Version 1.0.0

Description Convert text into target classifications (e.g., ISO 3166-1) using a JSON mapping with regular expressions. Provides helpers to return the full mapping and associated metadata.

Language en-US

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.2)

Imports jsonlite

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/econcz/rconvertu

BugReports https://github.com/econcz/rconvertu/issues

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Ilya Bolotov [aut, cre] (ORCID: <https://orcid.org/0000-0003-1148-7144>)

Maintainer Ilya Bolotov <ilya.bolotov@vse.cz>

Repository CRAN

Date/Publication 2026-05-16 22:40:02 UTC

Contents

cconv *Convert text into target classifications (e.g., ISO 3166-1) using a JSON mapping with regular expressions.*

Description

Pure-R implementation of the **convertu** API. Converts text into a target classification using a JSON mapping, or returns mapping/metadata (`info` / `dump` modes).

Usage

```
cconv(
  data = NULL,
  json_file = NULL,
  info = FALSE,
  dump = FALSE,
  to = NULL,
  text = character()
)

convertu(
  data = NULL,
  json_file = NULL,
  info = FALSE,
  dump = FALSE,
  to = NULL,
  text = character()
)
```

Arguments

`data`	list of named lists (optional). A complete classification mapping provided directly. If supplied without `json_file`, this data will be used in-memory for conversions without reading from disk. If both `data` and `json_file` are supplied, the data is written to `json_file` and the file path is returned.
`json_file`	character(1). Path to the classification JSON file. If not provided, the default bundled `classification.json` is used (resolved via `system.file("extdata", "classification.json", package="rconvertu")`). When `data` is not supplied, this file is loaded and used as the source mapping. When `data` is supplied along with `json_file`, the data is written to `json_file`.
`info`	logical(1). If `TRUE`, return only metadata/sources entries. No conversion is performed.
`dump`	logical(1). If `TRUE`, return the full mapping (filtered of metadata/sources). No conversion is performed.
`to`	character(1). Target field name to return from matched records (e.g., `"iso3"`).
`text`	character(). One or more input strings to convert. A single string input yields a single string output; a vector yields a character vector of converted results.

Details

Behavior:

- `info = TRUE` → returns only metadata and sources entries (no conversion).
- `dump = TRUE` → returns the full classification (no metadata/sources).
- Otherwise → converts `text` using regex-based matching and returns the value from the requested field `to`.

Value

If `info = TRUE` or `dump = TRUE`, returns a list of records. Otherwise, returns a character vector of converted values:

- If `length(text) == 1`, returns a length-one character scalar.
- If no match is found for an input, the original value is returned.

Data template (list of named lists)

The classification is a top-level list with three kinds of elements:

1. Many record elements (unnamed or named) with fields:
   - `regex` (chr): pattern matching the input text.
   - `name_en` (chr): English short name.
   - `name_fr` (chr): French short name (optional).
   - `iso3` (chr): alpha-3 code (example field).
   - `iso2` (chr): alpha-2 code (example field).
   - `isoN` (chr): numeric code (example field).
2. One element `metadata` (named list) mapping field names to their human-readable descriptions:

```
metadata = list(
  name_en = "English short name",
  name_fr = "French short name",
  iso3    = "alpha-3 code",
  iso2    = "alpha-2 code",
  isoN    = "numeric"
)
```

3. One element `sources` (character vector) with references:

```
sources = c(
  "https://www.iso.org/iso-3166-country-codes.html",
  "https://en.wikipedia.org/wiki/List_of_alternative_country_names"
)
```

Examples

```
# Single conversion
cconv(to = "iso3", text = "Czech Republic")

# Multiple conversions
cconv(to = "iso3", text = c("Czech Republic", "Slovakia"))

# Show bundled metadata
cconv(info = TRUE)

# Dump classification mapping only
cconv(dump = TRUE)
```

`check_classification` *Validate a classification list.*

Description

Ensures the provided JSON-based classification data follows the expected structure and contains valid fields.

Usage

```
check_classification(x)
```

Arguments

`x`	list. The classification object loaded via `jsonlite::fromJSON(..., simplifyVector = FALSE)`.

Value

Logical, `TRUE` if valid, otherwise an error is raised.

Examples

```
path <- system.file("extdata", "classification.json", package = "rconvertu")
cls <- jsonlite::fromJSON(path, simplifyVector = FALSE)
check_classification(cls)
```

Index