

# Package ‘and’

May 7, 2026

**Title** Construct Natural-Language Lists with Internationalization

**Version** 0.1.8

**Description** Construct language-aware lists. Make ``and"-separated and ``or"-separated lists that automatically conform to the user's language settings.

**License** MIT + file LICENSE

**URL** <https://pkg.rossellhayes.com/and/>,  
<https://github.com/rossellhayes/and>

**BugReports** <https://github.com/rossellhayes/and/issues>

**Depends** R (>= 3.5)

**Imports** glue, rlang (>= 1.0.0)

**Suggests** covr, knitr, mockery, stringi, testthat (>= 3.0.0), withr

**Config/testthat/edition** 3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** Alexander Rossell Hayes [aut, cre, cph] (ORCID:  
<<https://orcid.org/0000-0001-9412-0457>>),  
Unicode, Inc. [dct] (Language data),  
Flavia Rossell Hayes [ill]

**Maintainer** Alexander Rossell Hayes <[alexander@rossellhayes.com](mailto:alexander@rossellhayes.com)>

**Repository** CRAN

**Date/Publication** 2026-03-17 08:31:13 UTC

## Contents

and . . . . .	2
and_languages . . . . .	3
set_language . . . . .	4

---

and *Combine a vector into a natural language string*

---

### Description

These functions transform a vector into a single string similar to `knitr::combine_words()` or `glue::glue_collapse()`.

`and()` and `or()` natively support localization, using translations and punctuation to match the users' language settings. See [and\\_languages](#) for available languages.

- `and()` combines words using the native conjunctive ("and" in English)
- `or()` combines words using the native disjunctive ("or" in English)

### Usage

```
and(x, ..., language = NULL)
```

```
or(x, ..., language = NULL)
```

### Arguments

x	A list of <a href="#">character</a> strings to combine
...	These dots are for future extensions and must be empty.
language	The language to use for translation. If <code>NULL</code> , the default, the language of the user's R session is used.  Codes should be two or three lowercase letters representing the language, optionally followed by an underscore and two uppercase letters representing a territory. For example, "es" represents Spanish and "en_US" represents American English.  If a territory is specified but there is no specific translation for that territory, translations fall back to the general language. For example, <code>and</code> does not include specific translations for Canadian French, so "fr_CA" falls back to "fr".  If a language is specified that is not supported by <code>and</code> , translations fall back to English. For a list of supported languages, see <a href="#">and_languages</a> .

### Value

A [character](#) string of length 1

### Source

Language data is derived from the [Unicode Common Locale Data Repository \(CLDR\)](#)

**Examples**

```
and(1:3)
or(1:3)

and(1:3, language = "es")
or(1:3, language = "ja")
```

---

and_languages	<i>Supported languages</i>
---------------	----------------------------

---

**Description**

A list of supported languages and examples of their usage.

**Usage**

```
and_languages
```

**Format**

A data frame with 6 variables:

**language** The name of the language, possibly with a territory in parentheses

**code** The language code

**example\_and\_2** An example of a conjunctive list with two elements in the language

**example\_and\_4** An example of a conjunctive list with four elements in the language

**example\_or\_2** An example of a disjunctive list with two elements in the language

**example\_or\_4** An example of a disjunctive list with four elements in the language

**support** Either "full" or "partial". Partially supported languages generally localize `and()` but not `or()`.

**Source**

Language data is derived from the [Unicode Common Locale Data Repository \(CLDR\)](#)

**Examples**

```
and_languages
```

---

`set_language`*Change the language of the current R environment*

---

**Description**

Changes the value of the LANGUAGE environment variable.

Returns the value of the LANGUAGE environment variable before it was changed. This allows you to use the following structure to temporarily change the language:

```
old_language <- set_language("es")
on.exit(set_language(old_language))
```

**Usage**

```
set_language(language)
```

**Arguments**

language	A language code. Codes should be two or three lowercase letters representing the language, optionally followed by an underscore and two uppercase letters representing a territory. For example, "es" represents Spanish and "en_US" represents American English. If a territory is specified but there is no specific translation for that territory, translations fall back to the general language. For example, if there are no specific translations for Canadian French, "fr_CA" will fall back to "fr". If a language is specified but there is no translation for that language, translations generally fall back to English. If language is an empty string or NULL, the LANGUAGE environment variable is unset.
----------	---

**Value**

Returns the pre-existing value of the LANGUAGE environment variable

**Examples**

```
# Change language to Korean
set_language("ko")

# Change language to Mexican Spanish, which may fall back to "es"
set_language("es_MX")

# Temporarily set the language to Cantonese
old_language <- set_language("yue")
set_language(old_language)
```

```
# Change to an invalid language, which generally falls back to English  
set_language("zxx")
```

```
# Unset the language environment variable  
set_language(NULL)
```

# Index

## \* datasets

and\_languages, 3

and, 2

and(), 3

and\_languages, 2, 3

character, 2

glue::glue\_collapse(), 2

knitr::combine\_words(), 2

NULL, 2

or (and), 2

or(), 3

set\_language, 4