Package ‘Unicode’

May 18, 2025

Version 16.0.0-1

Encoding UTF-8

Title Unicode Data and Utilities

Description Data from Unicode 16.0.0 and related utilities.

Depends R (>= 3.5.0)

Imports utils

License GPL-2

NeedsCompilation no

Author Kurt Hornik [aut, cre] (ORCID: <`https://orcid.org/0000-0003-4198-9911`>)

Maintainer Kurt Hornik <`Kurt.Hornik@R-project.org`>

Repository CRAN

Date/Publication 2025-05-18 16:08:01 UTC

Contents

casefuns *Unicode Case Conversions*

Description

Default Unicode algorithms for case conversion.

Usage

```
u_to_lower_case(x)
u_to_upper_case(x)
u_to_title_case(x)
u_case_fold(x)
```

Arguments

x	R objects (see **Details**).

Details

These functions are generic functions, with methods for the Unicode character classes (u_char, u_char_range, and u_char_seq) which suitably apply the case mappings to the Unicode characters given by x, and a default method which treats x as a vector of “Unicode strings”, and returns a vector of UTF-8 encoded character strings with the results of the case conversion of the elements of x.

Currently, only the unconditional case maps are available for conversion to lower, upper or title case: other variants may be added eventually.

Currently, conversion to title case is only available for u_char objects. Other methods will be added eventually (once the Unicode text segmentation algorithm is implemented for detecting word boundaries).

Currently, u_case_fold only performs *full* case folding using the Unicode case mappings with status “C” and “F”: other variants will be added eventually.

Value

For the methods for the Unicode character classes, a u_char_seq vector of Unicode character sequences with the conversions of the characters in x.

For the default method, a UTF-8 encoded character string with the results of the case conversions of the elements of x.

Examples

```
## Latin upper case letters A to Z:
x <- as.u_char(as.u_char_range("0041..005A"))
## In case we did not know the code points, we could use e.g.
x <- as.u_char(utf8ToInt(paste(LETTERS, collapse = "")))
vapply(x, intToUtf8, "")
## Unicode character method:
```

```
vapply(u_to_lower_case(x), intToUtf8, "")
## Default method:
u_to_lower_case(LETTERS)

u_case_fold("Hi Dave.")

## More interesting stuff: sharp s.
u_to_upper_case("heiß")
## Note that the default full upper case mapping of U+00DF (LATIN SMALL
## LETTER SHARP S) is *not* to U+1E9E (LATIN CAPITAL LETTER SHARP S).
u_case_fold("heiß")
```

---

n_of_u_chars *Unicode Character Counts*

Description

Compute the number of Unicode characters (code points) in sequences or ranges of Unicode characters.

Usage

```
n_of_u_chars(x)
```

Arguments

x	a vector of Unicode characters, character ranges, or character sequences.

Value

An integer vector with the numbers of Unicode characters specified by the elements of x.

Examples

```
## How many code points are assigned to the Latin and Cyrillic scripts?
x <- u_scripts(c("Latn", "Cyrl"))
## Numbers in the respective ranges:
n <- lapply(x, n_of_u_chars)
n
## Total number:
vapply(n, sum, 0)
```

`tokenizers` *Unicode Alphabetic Tokenizer*

Description

A simple Unicode alphabetic tokenizer.

Usage

```
Unicode_alphabetic_tokenizer(x)
```

Arguments

`x`	a character vector.

Details

Tokenization first replaces the elements of `x` by their Unicode character sequences. Then, the non-alphabetic characters (i.e., the ones which do not have the Alphabetic property) are replaced by blanks, and the corresponding strings are split according to the blanks.

Value

A character vector with the tokenized strings.

`u_blocks` *Unicode Blocks*

Description

Unicode blocks.

Usage

```
u_blocks(x)
```

Arguments

`x`	a character vector with the names of Unicode blocks.

Value

If `x` is missing, a list of the Unicode blocks given as `u_char_range` Unicode character ranges, with the (full) block names as names.

If `x` is given, a (sub)list of the specific Unicode blocks.

References

Unicode Character Database (https://www.unicode.org/ucd/)

See Also

`u_char_property` to find the block (property) of Unicode characters.

`u_char_basics` *Unicode Character Objects*

Description

Data structures and basic methods for Unicode character data.

Usage

```
as.u_char(x)
as.u_char_range(x)
as.u_char_seq(x, sep = NA_character_)
```

Arguments

`x`	R objects coercible to the respective Unicode character data types, see **Details**.
`sep`	a character string.

Details

Package **Unicode** provides three basic classes for representing Unicode characters: `u_char` for vectors of Unicode characters, `u_char_range` for vectors of Unicode character ranges, and `u_char_seq` for vectors of Unicode character sequences. Objects from these classes are created via the respective coercion functions.

`as.u_char` knows to coerce integers or hex strings (with or without a leading '`0x`' or the '`U+`' typically used for Unicode characters) giving the corresponding code points. It can also handle Unicode character ranges, flattening them out into the corresponding vector of Unicode characters. To "coerce" a UTF-8 encoded R character string to the corresponding Unicode character object, use coercion on the result of obtaining the integer code points via `utf8ToInt`.

`as.u_char_range` knows to coerce character strings of single Unicode characters or a Unicode range expression with the hex codes of two Unicode characters collapsed by '`..`' (currently, hardwired). It can also handle `u_char` objects, coercing them to ranges of single code points.

`as.u_char_seq` knows to coerce character strings with the hex codes of Unicode characters collapsed by a non-empty `sep`. The default corresponds to using '`,`' if the strings use surrounding angles, and '` `' otherwise. If `sep` is empty or has length zero, the character strings are used as is, re-encoded in UTF-8 if necessary, and mapped to the corresponding Unicode character sequences using `utf8ToInt`. `as.u_char_seq` can also handle Unicode character ranges (giving the corresponding flattened out Unicode character sequences), or lists of objects coercible to Unicode characters via `as.u_char`.

All classes currently have as.character, as.data.frame, c, format, print, rep, unique and [subscript methods. More methods will be added eventually.

Value

For as.u_char, a u_char object giving a vector of Unicode characters.

For as.u_char_range, a u_char_range object giving a vector of Unicode character ranges.

For as.u_char_seq, a u_char_seq object giving a vector of Unicode character sequences.

References

Unicode Character Database (https://www.unicode.org/ucd/),
https://en.wikipedia.org/wiki/Unicode

Examples

```
x <- as.u_char_range(c("00AA..00AC", "01CC"))
x
## Corresponding Unicode character sequence object:
as.u_char_seq(x)
## Corresponding Unicode character object with all code points:
as.u_char(x)
## Inspect all Unicode characters in the range:
u_char_inspect(x)

## Turning R character strings into the respective Unicode character
## sequences:
as.u_char_seq(c("Austria", "Trantor"), "")
## which can then be subscripted "as usual", e.g.:
x <- as.u_char_seq(c("Austria", "Trantor"), "")[[1L]][c(3L, 5L)]
x
## To reassemble the character strings:
intToUtf8(x)
```

---

u_char_inspect *Unicode Character Inspection*

Description

Inspect Unicode characters.

Usage

```
u_char_inspect(x)
```

Arguments

x	an R object which can be coerced to a u_char vector of Unicode characters via as.u_char.

Value

A data frame with variables `Code`, `Name` and `Char`, giving the code and name of the given characters and the R character vectors corresponding to the code points.

Examples

```
## Who has ever seen a capital sharp s?
x <- u_char_from_name(c("LATIN SMALL LETTER SHARP S",
                        "LATIN CAPITAL LETTER SHARP S"))
u_char_inspect(x)
## (Does this display anything useful?)
```

`u_char_match` *Unicode Character Matching*

Description

Match Unicode characters to Unicode character ranges.

Usage

```
u_char_match(x, table, nomatch = NA_integer_)
x %uin% table
```

Arguments

`x`	an R object which can be coerced to a `u_char` vector of Unicode characters via `as.u_char`.
`table`	an R object coercible to a `u_char_range` vector of Unicode character range via `as.u_char_range`.
`nomatch`	the value to be returned (after coercion to integer) in the case when no match is found.

Details

`u_char_match` returns a vector of the positions of the (first) matches of the Unicode characters given by `x` (after coercion via `as.u_char`) to the Unicode character ranges given by `table` (after coercion via `as.u_char_range`).

`%uin%` returns a logical vector indicating if there was a match or not.

u_char_names *Unicode Character Names*

Description

Find the names or labels of Unicode characters, or Unicode characters by their name.

Usage

```
u_char_name(x)
u_char_from_name(x, type = c("exact", "grep"), ...)
u_char_label(x)
```

Arguments

x	an R object which can be coerced to a u_char vector of Unicode characters via as.u_char for u_char_name and u_char_label; a character vector otherwise.
type	one of "exact" or "grep", or an abbreviation thereof.
...	arguments to be passed to grepl when using this for pattern matching.

Details

The Unicode Standard provides a convention for labeling code points that do not have character names (control, reserved, noncharacter, private-use and surrogate code points). These labels can be obtained by u_char_label.

By default, exact matching is used for finding Unicode characters by name. When type = "grep", grepl is used for matching x against the Unicode character names; for now, Hangul syllable and CJK Unified Ideograph names are ignored in this case.

Value

For u_char_name and u_char_label, a character vector with the names or labels, respectively, of the corresponding Unicode characters.

For u_char_from_name, a u_char object giving the Unicode characters with name exactly matching the given names.

Examples

```
x <- as.u_char(utf8ToInt("Austria"))
u_char_name(x)

## Derived Hangul syllable character names are also supported for
## finding characters by exact matching:
x <- u_char_name("0xAC00")
x
u_char_from_name(x)
```

```
## Find all Unicode characters with name matching 'DIGIT ONE'.
x <- u_char_from_name("\\bDIGIT ONE\\b", "g")
## And show their names.
u_char_name(x)
```

`u_char_properties` *Unicode Character Properties*

Description

Get the properties of Unicode characters.

Usage

```
u_char_info(x)
u_char_properties(x, which)
u_char_property(x, which)
```

Arguments

`x`	an R object which can be coerced to a `u_char` vector of Unicode characters via `as.u_char`.
`which`	a character vector or string (for `u_char_property`), respectively, with the possibly abbreviated names of Unicode properties.

Value

For `u_char_info`, a data frame with variables giving the Code (`Code`) and the 'basic' Unicode variables Name, General Category, Canonical Combining Class, Bidi Class, Decomposition, Numeric Value Decimal Digit, Numeric Value Digit, Numeric Value, Bidi Mirrored, Unicode 1 Name, ISO Comment, Simple Uppercase Mapping, Simple Lowercase Mapping, and Simple Titlecase Mapping, with names obtained by replacing white spaces by underscores (e.g., `Bidi_Class`.)

For `u_char_properties`, a data frame with the values of the specified properties, or, if no arguments were given, a character vector with the names of all currently available Unicode character properties.

For `u_char_property`, the values of the specified property.

Note

Currently, only the property values of a subset of all Unicode character properties can be obtained.

References

Unicode Character Database (`https://www.unicode.org/ucd/`)

Examples

```
## When was the Euro sign added to Unicode?
x <- u_char_from_name("EURO SIGN")
u_char_property(x, "Age")

## List the currently available Unicode character properties.
u_char_properties()
```

`u_named_sequences` *Unicode Named Sequences*

Description

Unicode named sequences.

Usage

```
u_named_sequences()
```

Value

A data frame with elements `Name` and `Sequence` giving the names and the corresponding Unicode character sequences.

`u_scripts` *Unicode Scripts*

Description

Unicode scripts.

Usage

```
u_scripts(x)
```

Arguments

`x`	a character vector with the names of Unicode scripts.

Value

If `x` is missing, a list of the Unicode scripts given as `u_char_range` Unicode character ranges, with the (full) block names as names.

If `x` is given, a (sub)list of the specific Unicode scripts.

References

Unicode Character Database (https://www.unicode.org/ucd/)

See Also

u_char_property to find the script (property) of Unicode characters.

Examples

```
scripts <- u_scripts()
names(scripts)
## Total number of code points assigned to the scripts:
sort(vapply(scripts, function(s) sum(n_of_u_chars(s)), 0),
     decreasing = TRUE)
```

Index