Package 'charcuterie'

May 8, 2026

Title Handle Strings as Vectors of Characters

Version 0.0.6

Description Creates a new chars class which looks like a string but is actually a vector of individual characters, making 'strings' iterable. This class enables vector operations on 'strings' such as reverse, sort, head, and set operations.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0), vctrs

VignetteBuilder knitr

Config/testthat/edition 3

Imports generics, utils

URL https://github.com/jonocarroll/charcuterie, https://jonocarroll.github.io/charcuterie/

BugReports https://github.com/jonocarroll/charcuterie/issues

NeedsCompilation no

Author Jonathan Carroll [aut, cre] (ORCID: <https://orcid.org/0000-0002-1404-5264>)

Maintainer Jonathan Carroll <rpkg@jcarroll.com.au>

Repository CRAN

Date/Publication 2024-11-07 21:40:02 UTC

Contents

`c.chars` *Combine* `chars` *Objects*

Description

Combine `chars` Objects

Usage

```
## S3 method for class 'chars'
c(...)
```

Arguments

`...`	`chars` objects.

Value

a larger `chars` object containing the combined elements of `...`.

Examples

```
c(chars("java"), chars("script"))
```

chars *Create a* chars *Object*

Description

Create a chars Object

Usage

```
chars(x, ...)
```

Arguments

x	object to convert to chars.
...	other options passed to methods.

Value

an object of class chars.

chars.character *Create a* chars *Object From a String*

Description

Create a chars Object From a String

Usage

```
## S3 method for class 'character'
chars(x, ...)
```

Arguments

x	string to convert to a chars object (length 1 only).
...	unused

Details

chars expects a single string as input. To create a list of these, consider lapply(strings, chars)

Value

an object of class chars, essentially splitting the string into individual characters

chars.default *Convert an Object to* chars

Description

Convert an Object to chars

Usage

```
## Default S3 method:
chars(x, ...)
```

Arguments

x	object to convert
...	other options

Value

.NotYetImplemented() error

count *Count Characters in a Chars Object*

Description

Count Characters in a Chars Object

Usage

```
count(x, value, ignore.case = FALSE)
```

Arguments

x	A vector of characters.
value	character (length 1) to count
ignore.case	should the letter case be ignored?

Value

integer, count of instances of value in x

Examples

```
count(chars("strawberry"), 3)
```

except *Elements of x Except Those in y*

Description

Does not treat the operation as a set.

Usage

```
except(x, y)
```

Arguments

x	larger vector.
y	smaller vector.

Value

elements of x not appearing in y.

Examples

```
except(c(1:5), 3)
except(chars("abcde"), "c")
except(chars("abracadabra"), "b")
```

format.chars *Format a* chars *Object*

Description

Format a chars Object

Usage

```
## S3 method for class 'chars'
format(x, ...)
```

Arguments

x	a chars object.
...	further arguments passed to or from other methods.

Value

a formatted chars object.

head.chars *Return the First Parts of a* chars *Object*

Description

Return the First Parts of a chars Object

Usage

```
## S3 method for class 'chars'
head(x, ...)
```

Arguments

x	a chars object.
...	further arguments passed to or from other methods.

Value

the first (n) elements of a chars object as a chars object.

Examples

```
head(chars("abcdefhgi"))

head(chars("javascript"), 4)
```

intersect.chars *Setwise Intersection of* chars *Objects*

Description

Setwise Intersection of chars Objects

Usage

```
## S3 method for class 'chars'
intersect(x, y, ...)
```

Arguments

x	a chars object.
y	a chars object or character vector.
...	further arguments passed to or from other methods.

Value

the setwise intersection of x and y.

Examples

```
union(chars("pine"), chars("apple"))
```

---

is_alnum *Is a Character a Letter OR a Number?*

Description

A combination of `is_letter()` and `is_number()`.

Usage

```
is_alnum(x)
```

Arguments

x	A vector of characters.

Value

A boolean vector indicating whether each element of x is a letter or a number.

Examples

```
is_alnum(chars("Lee7c0deR 4 L1fe"))

Filter(is_alnum, chars("2 B or !2 B"))
```

---

is_letter *Is a Character a Letter?*

Description

Compares against the values of `letters` (the English alphabet), ignoring case.

Usage

```
is_letter(x)
```

Arguments

x	A vector of characters.

Value

A boolean vector indicating whether each element of `x` is a letter (appears in `letters` ignoring case).

Examples

```
is_letter(chars("Lee7c0deR"))

Filter(is_letter, chars("w00t"))
```

`is_number` *Is a Character a Number?*

Description

Compares against the values of `0:9` (as a number).

Usage

```
is_number(x)
```

Arguments

`x`	A vector of characters.

Value

A boolean vector indicating whether each element of `x` is a number (appears in `0:9` as a number)

Examples

```
is_number(chars("Lee7c0deR"))

Filter(is_number, chars("w00t"))
```

is_punct *Is a Character Punctuation?*

Description

Compares against the regex group `[[:punct:]]`.

Usage

```
is_punct(x)
```

Arguments

x	A vector of characters.

Value

A boolean vector indicating whether each element of x is considered as punctuation.

Examples

```
is_punct(chars("I can haz?"))

Filter(Negate(is_punct), chars("abc,123;$*%?"))
```

print.chars *Print a* chars *Object*

Description

Print a chars Object

Usage

```
## S3 method for class 'chars'
print(x, ...)
```

Arguments

x	a chars object.
...	further arguments passed to or from other methods.

Value

x (invisibly), used to print to console.

rev.chars *Reverse Elements of a* chars *Object*

Description

Reverse Elements of a chars Object

Usage

```
## S3 method for class 'chars'
rev(x)
```

Arguments

x	a chars object

Value

a chars object with the elements reversed.

Examples

```
rev(chars("racecar"))

rev(chars("alphabet"))
```

setdiff.chars *Setwise Difference Between* chars *Objects*

Description

Setwise Difference Between chars Objects

Usage

```
## S3 method for class 'chars'
setdiff(x, y, ...)
```

Arguments

x	a chars object.
y	a chars object or character vector.
...	further arguments passed to or from other methods.

Value

the setwise difference of x and y.

Examples

```
setdiff(chars("javascript"), chars("script"))
```

`sort.chars` *Sort a* `chars` *Object*

Description

Sort a chars Object

Usage

```
## S3 method for class 'chars'
sort(x, decreasing = FALSE, ...)
```

Arguments

x	a chars object.
decreasing	logical. Should the sort be increasing or decreasing? Not available for partial sorting.
...	further arguments passed to or from other methods.

Value

a sorted chars object.

Examples

```
sort(chars("alphabet"))
```

string *Create a String From a* chars *Object*

Description

Create a String From a chars Object

Usage

```
string(x, collapse = "", ...)
```

Arguments

x	one or more chars objects.
collapse	an optional character string to separate the results. Not NA_character_.
...	other arguments passed to paste()

Value

a character (traditional R string) with the elements of x in a single value.

tail.chars *Return the Last Parts of a* chars *Object*

Description

Return the Last Parts of a chars Object

Usage

```
## S3 method for class 'chars'
tail(x, ...)
```

Arguments

x	a chars object.
...	further arguments passed to or from other methods.

Value

the last (n) elements of a chars object as a chars object.

Examples

```
tail(chars("javascript"))

tail(chars("abcdefghi"))
```

union.chars *Setwise Union of* chars *Objects*

Description

Setwise Union of chars Objects

Usage

```
## S3 method for class 'chars'
union(x, y, ...)
```

Arguments

x	a chars object.
y	a chars object or character vector.
...	further arguments passed to or from other methods.

Value

the setwise union of x and y.

Examples

```
union(chars("java"), chars("script"))
```

unique.chars *Extract Unique Elements of* chars *Objects.*

Description

Extract Unique Elements of chars Objects.

Usage

```
## S3 method for class 'chars'
unique(x, ...)
```

Arguments

x	a chars object.
...	further arguments passed to or from other methods.

Value

a chars object containing unique elements.

Examples

```
unique(chars("mississippi"))
```

`[.chars` *Extract or Replace Parts of a* `chars` *Object*

Description

Extract or Replace Parts of a `chars` Object

Usage

```
## S3 method for class 'chars'
x[...]
```

Arguments

`x`	a `chars` object.
`...`	further arguments passed to or from other methods.

Value

the extracted parts of a `chars` object, or a `chars` object with replacements performed.

Examples

```
s <- chars("censor")
s[2:5]
s[2:5] <- "X"
s
```

Index