Package 'sylly'

June 10, 2017

Type Package

Title Hyphenation and Syllable Counting for Text Analysis

Author m.eik michalke [aut, cre]

Maintainer m.eik michalke <meik.michalke@hhu.de>

Depends R (>= 3.0.0),methods

Description This is a native R implementation of the hyphenation algorithm used for 'TeX'/'LaTeX' and similar software, as proposed by Liang (1983). Mainly contains the function 'hyphen()' to be used for hyphenation/syllable counting of text objects. It was originally developed for and part of the 'koRpus' package, but later released as a separate package so it's lighter to have this particular functionality available for other packages. Support for various languages needs be added on-the-fly or by plugin packages, this package does not include any language specific data. Due to some restrictions on CRAN, the full package sources are only available from the project homepage. To ask for help, report bugs, request features, or discuss the development of the package, please subscribe to the koRpus-dev mailing list (<http://korpusml.reaktanz.de>).

License GPL (>= 3)

Encoding UTF-8

LazyLoad yes

URL https://reaktanz.de/?c=hacking&s=koRpus

BugReports https://github.com/unDocUMeantIt/sylly/issues

Version 0.1-1

Date 2017-06-10

Collate '00_environment.R'
'01_class_01_kRp.hyph.pat.R'
'01_class_02_kRp.hyphen.R'
'02_method_correct.R'

'02_method_hyphen.R'
'02_method_kRp.hyphen.R'
'02_method_show.kRp.hyphen.R'
'02_method_summary.kRp.hyphen.R'
'get.sylly.env.R'
'manage.hyph.pat.R'
'read.hyph.pat.R'
'set.hyph.support.R'
'set.sylly.env.R'
'sylly-internal.R'
'sylly-internal_langpack_generator.R'
'sylly-package.R'

RoxygenNote 6.0.1

R topics documented:

`sylly-package` *The sylly Package*

Description

Hyphenation and Syllable Counting for Text Analysis.

Details

Package:	sylly
Type:	Package
Version:	0.1-1
Date:	2017-06-10
Depends:	R (>= 3.0.0),methods

Encoding:	UTF-8
License:	GPL (>= 3)
LazyLoad:	yes
URL:	https://reaktanz.de/?c=hacking&s=koRpus

This is a native R implementation of the hyphenation algorithm used for 'TeX'/'LaTeX' and similar software, as proposed by Liang (1983). Mainly contains the function 'hyphen()' to be used for hyphenation/syllable counting of text objects. It was originally developed for and part of the 'koRpus' package, but later released as a separate package so it's lighter to have this particular functionality available for other packages. Support for various languages needs be added on-the-fly or by plugin packages, this package does not include any language specific data. Due to some restrictions on CRAN, the full package sources are only available from the project homepage. To ask for help, report bugs, request features, or discuss the development of the package, please subscribe to the koRpus-dev mailing list (<http://korpusml.reaktanz.de>).

Author(s)

m.eik michalke

`correct.hyph` *Correct kRp.hyphen objects*

Description

The method `correct.hyph` can be used to alter objects of class `kRp.hyphen-class`.

Usage

```
correct.hyph(obj, word = NULL, hyphen = NULL, cache = TRUE)

## S4 method for signature 'kRp.hyphen'
correct.hyph(obj, word = NULL, hyphen = NULL,
  cache = TRUE)
```

Arguments

`obj`	An object of class `kRp.hyphen-class`.
`word`	A character string, the (possibly incorrectly hyphenated) `word` entry to be replaced with `hyphen`.
`hyphen`	A character string, the new manually hyphenated version of `word`. Mustn't contain anything other than characters of `word` plus the hyphenation mark `"-"`.
`cache`	Logical, if `TRUE`, the given hyphenation will be added to the sessions' hyphenation cache. Existing entries for the same word will be replaced.

Details

Although hyphenation should turn out to be rather accurate, the algorithm does ususally produce some errors. If you want to correct for these flaws, this method can be of help, because it might prevent you from introducing new errors. That is, it will do some sanitiy checks before the object is actually manipulated and returned.

That is, `correct.hyph` checks whether `word` and `hyphen` are actually hyphenations of the same token before proceeding. If so, it will also recalculate the number of syllables and update the `syll` field.

If both `word` and `hyphen` are `NULL`, `correct.hyph` will try to simply recalculate the syllable count for each word, by counting the hyphenation marks (and adding 1 to the number). This can be usefull if you changed hyphenation some other way, e.g. in a spreadsheet GUI, but don't want to have to correct the syllable count yourself as well.

Value

An object of the same class as `obj`.

Examples

```
## Not run:
hyphenated.txt <- correct.hyph(hyphenated.txt, "Hilfe", "Hil-fe")

## End(Not run)
```

`describe` *Getter/setter methods for sylly objects*

Description

These methods should be used to get or set values of hyphenated text objects generated by functions like `hyphen()`.

Usage

```
describe(obj)

## S4 method for signature 'kRp.hyphen'
describe(obj)

describe(obj) <- value

## S4 replacement method for signature 'kRp.hyphen'
describe(obj) <- value

language(obj)
```

```
## S4 method for signature 'kRp.hyphen'
language(obj)

language(obj) <- value

## S4 replacement method for signature 'kRp.hyphen'
language(obj) <- value

hyphenText(obj)

## S4 method for signature 'kRp.hyphen'
hyphenText(obj)

hyphenText(obj) <- value

## S4 replacement method for signature 'kRp.hyphen'
hyphenText(obj) <- value

## S4 method for signature 'kRp.hyphen'
x[i, j]

## S4 replacement method for signature 'kRp.hyphen'
x[i, j] <- value

## S4 method for signature 'kRp.hyphen'
x[[i]]

## S4 replacement method for signature 'kRp.hyphen'
x[[i]] <- value
```

Arguments

obj	An object of class kRp.hyphen.
value	A value to set.
x	An object of class kRp.hyphen.
i	Row index.
j	Column index.

Details

- describe() returns the desc slot.
- language() returns the lang slot.
- hyphenText() returns the hyphen slot from objects of class kRp.hyphen.
- [/[[Can be used as a shortcut to index the results of hyphenText().

Examples

```
## Not run:
hyphenText(hyphenated.txt)

## End(Not run)
```

`get.sylly.env` *Get sylly session environment*

Description

The function `get.sylly.env` returns information on your session environment regarding the sylly package, e.g. whether a cache file should be used, if it was set before using `set.sylly.env`.

Usage

```
get.sylly.env(..., errorIfUnset = TRUE)
```

Arguments

`...`	Named parameters to get from the sylly environment. Valid arguments are: **lang** Logical, whether the set language should be returned. **hyph.cache.file** Logical, whether the set hyphenation cache file for hyphen should be returned. **hyph.max.token.length** Logical, whether the set maximum token length should be returned.
`errorIfUnset`	Logical, if `TRUE` and the desired property is not set at all, the function will fail with an error message.

Value

A character string or list, possibly including:

`lang`	The specified language
`hyph.cache.file`	The specified hyphenation cache file for hyphen

See Also

`set.sylly.env`

Examples

```
## Not run:
set.sylly.env(hyph.cache.file="/tmp/cache_file.RData")
get.sylly.env(hyph.cache.file=TRUE)

## End(Not run)
```

hyphen *Automatic hyphenation*

Description

These methods implement word hyphenation, based on Liang's algorithm.

Usage

```
hyphen(words, ...)

## S4 method for signature 'character'
hyphen(words, hyph.pattern = NULL, min.length = 4,
  rm.hyph = TRUE, quiet = FALSE, cache = TRUE, as = "kRp.hyphen")

hyphen_df(words, ...)

## S4 method for signature 'character'
hyphen_df(words, hyph.pattern = NULL, min.length = 4,
  rm.hyph = TRUE, quiet = FALSE, cache = TRUE)

hyphen_c(words, ...)

## S4 method for signature 'character'
hyphen_c(words, hyph.pattern = NULL, min.length = 4,
  rm.hyph = TRUE, quiet = FALSE, cache = TRUE)
```

Arguments

words	Either a character vector with words/tokens to be hyphenated, or any tagged text object generated with the koRpus package.
...	Only used for the method generic.
hyph.pattern	Either an object of class kRp.hyph.pat-class, or a valid character string naming the language of the patterns to be used (must already be loaded, see details).
min.length	Integer, number of letters a word must have for considering a hyphenation. hyphen will not split words after the first or before the last letter, so values smaller than 4 are not useful.
rm.hyph	Logical, whether appearing hyphens in words should be removed before pattern matching.
quiet	Logical. If FALSE, short status messages will be shown.
cache	Logical. hyphen() can cache results to speed up the process. If this option is set to TRUE, the current cache will be queried and new tokens also be added. Caches are language-specific and reside in an environment, i.e., they are cleaned at the end of a session. If you want to save these for later use, see the option hyph.cache.file in set.sylly.env.

as — A character string defining the class of the object to be returned. Defaults to "kRp.hyphen", but can also be set to "data.frame" or "numeric", returning only the central data.frame or the numeric vector of counted syllables, respectively. For the latter two options, you can alternatively use the shortcut methods hyphen_df or hyphen_c.

Details

For this to work the function must be told which pattern set it should use to find the right hyphenation spots. The most straight forward way to add suuport for a particular language during a session is to load the appropriate language package (e.g., the package sylly.en for English or sylly.de for German).

After such a package was loaded, you can simply use the language abbreviation as the value for the hyph.pattern argument (like "en" for the English pattern set). If words is an object that was tokenized and tagged with the koRpus package, its language definition can be used instead, i.e. you don't need to specify hyph.pattern, hyphen will pick the language automatically.

In case you'd rather use your own pattern set, hyph.pattern can be an object of class kRp.hyph.pat, alternatively.

Value

An object of class kRp.hyphen-class, data.frame or a numeric vector, depending on the value of the as argument.

References

Liang, F.M. (1983). *Word Hy-phen-a-tion by Com-put-er*. Dissertation, Stanford University, Dept. of Computer Science.

See Also

read.hyph.pat, manage.hyph.pat

Examples

```
## Not run:
library(sylly.en)
sampleText <- c("This", "is", "a", "rather", "stupid", "demonstration")
hyphen(sampleText, hyph.pattern="en")
hyphen_df(sampleText, hyph.pattern="en")
hyphen_c(sampleText, hyph.pattern="en")

# using a koRpus object
hyphen(tagged.text)

## End(Not run)
```

kRp.hyphen,-class *S4 Class kRp.hyphen*

Description

This class is used for objects that are returned by hyphen.

Slots

`lang` A character string, naming the language that is assumed for the analized text in this object

`desc` Descriptive statistics of the analyzed text.

`hyphen` A data.frame with two columns:

- `syll`: Number of recognized syllables
- `word`: The hyphenated word

kRp.hyph.pat,-class *S4 Class kRp.hyph.pat*

Description

This class is used for objects that are returned by `read.hyph.pat`.

Details

Since this package has been a part of the koRpus package before, you might run into old pattern files. You will know that this is the case if using them automatically tries to load the koRpus package. In these cases, you might want to strip the defunct reference to koRpus by calling the private function `sylly:::koRpus2sylly` which take the path to the old file as its first argument. Be aware that calling this function will overwrite the old file in-place, so you should make a backup first!

Slots

`lang` A character string, naming the language that is assumed for the patterns in this object

`pattern` A matrix with three colums:

- `orig`: The unchanged patgen patterns.
- `char`: Only the characters used for matching.
- `nums`: The hyphenation number code for the pattern.

`manage.hyph.pat` *Handling hyphenation pattern objects*

Description

This function can be used to examine and change hyphenation pattern objects be used with hyphen.

Usage

```
manage.hyph.pat(hyph.pattern, get = NULL, set = NULL, rm = NULL,
  word = NULL, min.length = 3L, rm.hyph = TRUE)
```

Arguments

`hyph.pattern`	Either an object of class `kRp.hyph.pat`, or a valid language abbreviation for patterns included in this package.
`get`	A character string, part of a word to look up in the pattern set, i.e., without the numbers indicating split probability.
`set`	A character string, a full pattern to be added to the pattern set, i.e., including the numbers indicating split probability.
`rm`	A character string, part of a word to remove from the pattern set, i.e., without the numbers indicating split probability.
`word`	A character string, a full word to hyphenate using the given pattern set.
`min.length`	Integer, number of letters a word must have for considering a hyphenation.
`rm.hyph`	Logical, whether appearing hyphens in words should be removed before pattern matching.

Details

You can only run one of the possible actions at a time. If any of these arguments is not `NULL`, the corresponding action is done in the following order, and every additional discarded:

- `get`Searches the pattern set for a given word part
- `set`Adds or replaces a pattern in the set (duplicates are removed)
- `rm`Removes a word part and its pattern from the set
- `word`Hyphenates a word and returns all parts examined as well as all matching patterns

If all action arguments are `NULL`, `manage.hyph.pat` returns the full pattern object.

Value

If all action arguments are `NULL`, returns an object of class `kRp.hyph.pat-class`. The same is true if `set` or `rm` are set and `hyph.pattern` is itself an object of that class; if you refer to a language instead, pattern changes will be done internally for the running session and take effect immediately. The `get` argument will return a caracter vector, and `word` a data frame.

References

[1] http://tug.ctan.org/tex-archive/language/hyph-utf8/tex/generic/hyph-utf8/patterns/txt/

See Also

kRp.hyph.pat-class, hyphen

Examples

```
## Not run:
manage.hyph.pat("en", set="r3ticl")
manage.hyph.pat("en", get="rticl")
manage.hyph.pat("en", word="article")
manage.hyph.pat("en", rm="rticl")

## End(Not run)
```

---

read.hyph.pat *Reading patgen-compatible hyphenation pattern files*

Description

This function reads hyphenation pattern files, to be used with hyphen.

Usage

```
read.hyph.pat(file, lang, fileEncoding = "UTF-8")
```

Arguments

file	A connection or character string with a valid path to a file with hyphenation patterns (one pattern per line).
lang	A character string, usually two letters short, naming the language the patterns are meant to be used with (e.g. "es" for Spanish).
fileEncoding	A character string defining the character encoding of the file to be read. Unless you have a really good reason to do otherwise, your pattern files should all be UTF-8 encoded.

Details

Hyphenation patterns that can be used are available from CTAN[1]. But actually any file with only the patterns themselves, one per line, should work.

The language designation is of no direct consequence here, but if the resulting pattern object is to be used by other functions in this package or koRpus, it should resamble the designation that's used for the same language there.

Value

An object of class kRp.hyph.pat-class.

References

[1] http://tug.ctan.org/tex-archive/language/hyph-utf8/tex/generic/hyph-utf8/patterns/txt/

See Also

hyphen, manage.hyph.pat

Examples

```
## Not run:
read.hyph.pat("~/patterns/hyph-en-us.pat.txt", lang="en_us")

## End(Not run)
```

---

set.hyph.support *Add support for new languages*

Description

You can use this function to add new languages to be used with sylly.

Usage

```
set.hyph.support(value)
```

Arguments

value	A named list that upholds exactly the structure defined above.

Details

Language support in this package is designed to be extended easily. You could call it modular, although it's actually more "environemntal", but nevermind.

To add new language support, say for Xyzedish, you basically have to call this function once and provide respective hyphenation patterns. If you would like to re-use this language support, you should consider making it a package.

If it succeeds, it will fill an internal environment with the information you have defined. hyphen will then know which language patterns are available as data files (which you must provide also).

You provide the meta data as a named list. It usually has one single entry to tell the new language abbreviation, e.g., set.hyph.support(list("xyz"="xyz")). However, this will only work if a) the language support script is a part of the sylly package itself, and b) the hyphen pattern is located in its data subdirectory.

For your custom hyphenation patterns to be found automatically, provide it as the value in the named list, e.g., `set.hyph.support(list("xyz"=hyph.xyz))`. This will directly add the patterns to `sylly`'s environment, so it will be found when hyphenation is requested for language `"xyz"`.

If you would like to provide hyphenation as part of a third party language package, you must name the object `hyph.<lang>`, save it to your package's `data` subdirectory named `hyph.<lang>.rda`, and append `package="<yourpackage>"` to the named list; e.g., `set.hyph.support(list("xyz"=c("xyz", package="koRpus`
Only then `sylly` will look for the pattern object in your package, not its own `data` directory.

Hyphenation patterns

To be able to also do syllable count with the newly added language, you should add a hyphenation pattern file as well. Refer to the documentation of read.hyph.pat() to learn how to produce a pattern object from a downloaded hyphenation pattern file. Make sure you use the correct name scheme (e.g. "hyph.xyz.rda") and good compression.

Examples

```
## Not run:
set.hyph.support(
  list("xyz"="xyz")
)

## End(Not run)
```

---

set.sylly.env *A function to set information on your sylly environment*

Description

The function `set.sylly.env` can be called before any of the hyphenation functions. It preserves some information on your current session's settings to a hidden environment.

Usage

```
set.sylly.env(..., validate = TRUE)
```

Arguments

`...`	Named parameters to set in the sylly environment. Valid arguments are: **lang** A character string specifying a valid language. **hyph.cache.file** A character string specifying a path to a file to use for storing already hyphenated data, used by hyphen. **hyph.max.token.length** A single number to set the internal cache size for tokens. The value should be set to the longest token to be hyphenated. To explicitly unset a value again, set it to an empty character string (e.g., `lang=""`).
`validate`	Logical, if `TRUE` given paths will be checked for actual availablity, and the function will fail if files can't be found.

Details

To get the contents of the hidden environment, the function `get.sylly.env` can be used.

Value

Returns an invisible `NULL`.

See Also

`get.sylly.env`

Examples

```
## Not run:
set.sylly.env(hyph.cache.file="/tmp/cache_file.RData")
get.sylly.env(hyph.cache.file=TRUE)

## End(Not run)
```

---

show,kRp.hyphen-method

Show method for sylly objects

Description

Show method for S4 objects of class `kRp.hyphen-class`.

Usage

```
## S4 method for signature 'kRp.hyphen'
show(object)
```

Arguments

`object`	An object of class `kRp.hyphen`.

See Also

`kRp.hyphen-class`

Examples

```
## Not run:
  hyphen(tagged.text)

## End(Not run)
```

summary,kRp.hyphen-method *Summary method for sylly objects*

Description

Summary method for S4 objects of class kRp.hyphen-class.

Usage

```
## S4 method for signature 'kRp.hyphen'
summary(object)
```

Arguments

object	An object of class kRp.hyphen.

See Also

kRp.hyphen-class

Examples

```
## Not run:
summary(hyphen(tagged.text))

## End(Not run)
```

Index