

Package ‘rcqp’

March 7, 2018

Type Package

Title Interface to the Corpus Query Protocol

Version 0.5

Date 2018-02-23

Author Bernard Desgraupes <bernard.desgraupes@u-paris10.fr>, Sylvain
Loiseau <sylvain.loiseau@univ-paris13.fr>

Maintainer Bernard Desgraupes <bernard.desgraupes@u-paris10.fr>

Depends R (>= 1.8.0), plyr

Suggests reshape

Description Implements Corpus Query Protocol functions based on the
CWB software. Rely on CWB (GPL v2), PCRE (BSD licence), glib2
(LGPL).

License GPL-2 | file LICENCE

LazyLoad yes

URL <https://www.r-project.org>

Collate main.R zzz.R s3.R

Encoding latin1

SystemRequirements GNU make, pcre (>= 7), glib2 (>= 2)

OS_type unix

R topics documented:

corpus	2
cqi_attributes	3
cqi_attribute_size	4
cqi_corpus_info	5
cqi_cpos2id	6
cqi_dump_subcorpus	8
cqi_fdist	9
cqi_full_name	10
cqi_id2cpos	11
cqi_list_corpora	13
cqi_list_subcorpora	14
cqi_query	15

cqi_regex2id	16
cqi_struct2cpos	17
cqp_flist	19
cqp_ftable	20
cqp_kwic	22
cqp_registry	23
Extraction	24
print.cqp_flist	25
rcqp	26
region_sizes	28
size	29
sort.cqp_kwic	30
subcorpus	31
summary.cqp_flist	32
Type and token	33
write.cqp_corpus	34

Index	36
--------------	-----------

corpus	<i>Create a corpus object</i>
--------	-------------------------------

Description

Create a corpus object wrapping a CWB corpus object.

Usage

```
corpus(corpus.name)
```

Arguments

corpus.name The name of a corpus in the CWB registry (must be uppercase).

Value

A corpus object.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[subcorpus](#), [print.cqp_corpus](#), [summary.cqp_corpus](#), [write.cqp_corpus](#), [cqp_kwic](#).

Examples

```
## Not run:  
c <- corpus("DICKENS");  
  
## End(Not run)
```

cqi_attributes

List the available attributes

Description

Get the list of the currently defined attributes in a given corpus.

Usage

```
cqi_attributes(corpus, type)
```

Arguments

corpus	(<i>string</i>) the name of the corpus (in uppercase letters).
type	(<i>character</i>) a character designating a type of attributes. Possible values are "a" for alignment attributes, "p" for positional attributes and "s" for structural attributes.

Details

For a specified corpus, this function lists the names of the different kinds of attributes used for the indexation. There are three kinds of attributes: positional, structural and alignment attributes (for aligned corpora).

Value

cqi_attributes returns a list of strings which are the names of the requested attributes.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_attribute_size](#), [cqi_structural_attribute_has_values](#).

Examples

```
## Not run:
cqi_attributes("DICKENS", "p")
cqi_attributes("DICKENS", "s")
cqi_attributes("DICKENS", "a")

## End(Not run)
```

cqi_attribute_size	<i>Get the size of an attribute</i>
--------------------	-------------------------------------

Description

Find the number of items (type/category or token/individual) of an attribute.

Usage

```
cqi_attribute_size(attribute)
cqi_structural_attribute_has_values(attribute)
cqi_lexicon_size(attribute)
```

Arguments

attribute (*string*) the qualified name of an attribute.

Details

The `cqi_attribute_size` indicates the number of actual elements (number of occurrences) of an attributes:

- on a positionnal attribute, it gives the number of tokens.
- on a structural attribute, it gives the number of regions.
- on an alignment attribute, it gives the number of aligned pairs.

The `cqi_lexicon_size` function indicates the number of different values of a positional attribute (number of types, or categories). It applies only to positional attributes.

The `cqi_structural_attribute_has_values` indicates if the specified attribute does have values. This concerns only the structural attributes. Positional attributes always have values. In order to get the value of a positionnal attribute with value, consider using [cqi_struc2str](#).

The attribute argument is the qualified name of an attribute. This concerns any kind of attribute. The syntax is of the form `name_of_corpus.name_of_attribute`. For instance *"DICKENS.lemma"* (positional attribute) or *"DICKENS.chapter"* (structural attribute).

Value

`cqi_attribute_size` returns an integer which is the number of different items in the corpus marked with the specified attribute.

`cqi_structural_attribute_has_values` returns a logical value telling whether the specified attribute has values (TRUE) or not (FALSE).

`cqi_lexicon_size` returns an integer value.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_attributes](#), [cqi_struct2str](#).

Examples

```
## Not run:
cqi_structural_attribute_has_values("DICKENS.file")
cqi_structural_attribute_has_values("DICKENS.file_name")
cqi_structural_attribute_has_values("DICKENS.chapter")
cqi_structural_attribute_has_values("DICKENS.chapter_title")

cqi_lexicon_size("DICKENS.lemma")
cqi_lexicon_size("DICKENS.pos")
cqi_lexicon_size("DICKENS.word")

# Positional attribute
cqi_attribute_size("DICKENS.lemma")
cqi_attribute_size("DICKENS.pos")
cqi_attribute_size("DICKENS.word")

# Structural attributes
cqi_attribute_size("DICKENS.book")
cqi_attribute_size("DICKENS.file")
cqi_attribute_size("DICKENS.chapter")

## End(Not run)
```

cqi_corpus_info

Info about a corpus

Description

Print info about all cwb files for a corpus

Usage

```
cqi_corpus_info(corpus)
```


Details

The `cqi_cpos2id` and `cqi_cpos2str` functions apply to positional attributes.

The `cqi_cpos2lbound`, `cqi_cpos2rbound` and `cqi_cpos2struc` functions apply to structural attributes.

The `cqi_alg2cpo` function applies to alignment attributes (in the case of aligned corpora).

Value

`cqi_cpos2str` returns a vector of strings.

The `cqi_alg2cpo` function returns a 4-elements vector representing the start and end positions of an alignment in the source corpus and the start and end positions in the target corpus.

The other functions return a vector of integers representing corpus positions.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.

Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_id2cpo](#), [cqi_id2freq](#), [cqi_id2str](#), [cqi_regex2id](#), [cqi_str2id](#), [cqi_struc2cpo](#), [cqi_struc2str](#).

Examples

```
## Not run:
cqi_query("DICKENS", "Foo", "\"eas(y|ier|iest)\"";)
cpo <- cqi_dump_subcorpus("DICKENS:Foo")[1:20,1]

cqi_cpos2id("DICKENS.pso", cpo)
cqi_cpos2str("DICKENS.pso", cpo)
cqi_cpos2id("DICKENS.word", cpo)
cqi_cpos2str("DICKENS.word", cpo)
cqi_cpos2id("DICKENS.lemma", cpo)
cqi_cpos2str("DICKENS.lemma", cpo)

cqi_query("DICKENS", "Foo", "\"interesting\"";)
cpo <- cqi_dump_subcorpus("DICKENS:Foo", 1, 10)[,1]

# Expects structural attributes
cqi_cpos2lbound("DICKENS.chapter", cpo)
cqi_cpos2rbound("DICKENS.chapter", cpo)

cqi_cpos2struc("DICKENS.chapter", cpo)

## End(Not run)
```

cqi_dump_subcorpus *Handle subcorpora*

Description

Get size and corpus positions corresponding to the results of a query.

Usage

```
cqi_dump_subcorpus(subcorpus, first=0, last=cqi_subcorpus_size(subcorpus)-1)
cqi_subcorpus_size(subcorpus)
cqi_drop_subcorpus(subcorpus)
```

Arguments

subcorpus	(<i>string</i>) qualified name of a subcorpus.
first	(<i>integer</i>) first index. Optional: default is 0.
last	(<i>integer</i>) last index. Optional: default is size-1.

Details

A subcorpus is typically the result of a previous query (see the [cqi_query](#) function). The qualified name of the subcorpus is of the form `name_of_corpus:name_of_subcorpus`, for instance `"DICKENS:A"`. The name of the subcorpus is declared in the second argument of the [cqi_query](#) function.

The `cqi_dump_subcorpus` function builds a two-dimension array (a matrix) of corpus positions corresponding to the results of a query.

The `cqi_drop_subcorpus` function frees the memory allocated to a subcorpus.

Value

`cqi_subcorpus_size` returns an integer which is the number of matches in the specified subcorpus.

`cqi_dump_subcorpus` returns a matrix with four columns corresponding to the *match*, *matchend*, *target*, and *keyword* fields respectively. Each row corresponds to a query match. The number of rows is `last-first+1`. This is the size of the subcorpus if the first and last arguments are not specified.

`cqi_drop_subcorpus` does not return anything.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_query](#), [cqi_list_corpora](#), [cqi_list_subcorpora](#).

Examples

```
## Not run:
cqi_query("DICKENS","Foo","\interesting\";")
cqi_subcorpus_size("DICKENS:Foo")
cqi_dump_subcorpus("DICKENS:Foo")
cqi_dump_subcorpus("DICKENS:Foo",4,10)

## End(Not run)
```

cqi_fdist

Frequency distributions

Description

Calculate a frequency list or a cross-tabulated frequency table.

Usage

```
cqi_fdist1(subcorpus, field1, key1, cutoff=0, offset=0)
cqi_fdist2(subcorpus, field1, key1, field2, key2, cutoff=0)
```

Arguments

subcorpus	(<i>string</i>) qualified name of a subcorpus.
field1	(<i>string</i>) the name of the anchor. It can be one of : <i>'match'</i> , <i>'matchend'</i> , <i>'target'</i> , <i>'keyword'</i> .
key1	(<i>string</i>) the name of a positional attribute.
field2	(<i>string</i>) the name of a second anchor. It can be one of : <i>'match'</i> , <i>'matchend'</i> , <i>'target'</i> , <i>'keyword'</i> .
key2	(<i>string</i>) the name of a positional attribute for the second anchor.
cutoff	(<i>integer</i>) a floor value under which results are not displayed. Default value is 0.
offset	(<i>integer</i>) an offset relative to the specified anchor.

Details

cqi_fdist1 builds a frequency list given individuals (occurrences) and modalities (a positional attribute).

The occurrences are defined by providing one of the anchors of a query (*'match'*, *'matchend'*, *'target'*, *'keyword'*).

The results are sorted in decreasing order of frequency. The cut argument specifies a value under which the results will not be returned. For instance, if the value is set to 10, only items with a frequency greater than or equal to 10 are returned. The default value of this argument is 0 which means that all the frequencies are returned by default.

The offset argument lets specify a position relative to the anchor specified by the field argument. For instance, if field is set to *'match'* and offset is equal to -1, the frequency list is computed on all the tokens located before the match anchor. The default value of this argument is 0.

cqi_fdist2 builds a frequency table of the values found in one anchor (such as *'match'*, *'matchend'*, *'target'*, *'keyword'*) cross-tabulated with the values found in another anchor. In other words, it gives the frequency of every different co-occurrences found according to the two given anchors.

Value

cqi_fdist1 returns a matrix with two columns. The first column contains the IDs of the attributes and the second column the corresponding frequency (number of occurrences).

cqi_fdist2 returns a matrix with three columns. The first column contains the IDs of an attribute of the occurrences found at the first anchor, the second column contains the IDs of an attribute of the occurrences found in the second anchor and the third column gives the frequency of the co-occurrences.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.

Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_list_subcorpora](#), [cqi_dump_subcorpus](#).

Examples

```
## Not run:
cqi_query("DICKENS", "Go", "[lemma=\"go\"] \"and\" [;]")
m <- cqi_fdist1("DICKENS:Go", "matchend", "pos")
cqi_id2str("DICKENS.pos", m[,1])

cqi_query("DICKENS", "NP", "[pos=\"DT\"] @[pos=\"JJ\"]? [pos=\"NNS?\";]")
cqi_fdist1("DICKENS:NP", "target", "lemma", 300)
cqi_fdist1("DICKENS:NP", "match", "lemma", cutoff=2000, offset=-1)

cqi_fdist2("DICKENS:Go", "matchend", "pos", "matchend", "lemma")
cqi_fdist2("DICKENS:NP", "target", "lemma", "matchend", "word", cutoff=300)

## End(Not run)
```

cqi_full_name

Full name of a corpus

Description

Get the descriptive string of a given corpus.

Usage

```
cqi_full_name(corpus)
```

Arguments

corpus (*string*) the name of a corpus (in uppercase letters).

Details

The descriptive string of a corpus is specified in the NAME parameter of the corpus declaration in the registry.

Value

A string.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_list_corpora](#).

Examples

```
## Not run:
cqi_full_name("DICKENS")

## End(Not run)
```

cqi_id2cpos

Convert the ID of an item

Description

Associate an item's ID to other parameters.

Usage

```
cqi_id2cpos(attribute, id)
cqi_id2freq(attribute, ids)
cqi_id2str(attribute, ids)
cqi_str2id(attribute, str)
```

Arguments

attribute	(<i>string</i>) the qualified name of an attribute.
id	(<i>integer</i>) a single value
ids	(<i>integer vector</i>)
strs	(<i>character vector</i>)

Details

These functions take as argument the ID of a positional attribute item and associate corresponding parameters such as the corpus position, the frequency or the string it represents.

The `cqi_str2id` function does the opposite: it associates the ID to a given positional parameter.

Value

The `cqi_id2cpos`, `cqi_id2freq`, `cqi_str2id` functions return an integer. The `cqi_id2str` function returns a string.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_alg2cpos](#) [cqi_cpos2alg](#), [cqi_cpos2id](#), [cqi_cpos2lbound](#), [cqi_cpos2rbound](#), [cqi_cpos2str](#),
[cqi_cpos2struc](#), [cqi_regex2id](#), [cqi_str2id](#), [cqi_struc2cpos](#), [cqi_struc2str](#).

Examples

```
## Not run:
cqi_query("DICKENS", "Foo", "\"eas(y|ier|iest)\"")
cpos <- cqi_dump_subcorpus("DICKENS:Foo")[1,1]

# # cqi_id2cpos
pids <- unique(cqi_cpos2id("DICKENS.pos", cpos))
ppos <- cqi_id2cpos("DICKENS.pos", pids[1])
ppos[1:50]

wids <- unique(cqi_cpos2id("DICKENS.word", cpos))
wpos <- cqi_id2cpos("DICKENS.word", wids[1])
wpos[1:50]

lids <- unique(cqi_cpos2id("DICKENS.lemma", cpos))
lpos <- cqi_id2cpos("DICKENS.lemma", lids[1])
lpos[1:50]
```

```
# # cqi_str2id and cqi_id2str
pid <- cqi_str2id("DICKENS.pos", "JJ")
cqi_id2str("DICKENS.pos", pid)

wid <- cqi_str2id("DICKENS.word", "easiest")
cqi_id2str("DICKENS.word", wid)

lid <- cqi_str2id("DICKENS.lemma", "easy")
cqi_id2str("DICKENS.lemma", lid)

# # cqi_id2freq
# Take the 20 first positions
cpos <- cqi_dump_subcorpus("DICKENS:Foo")[1:20,1]

pids <- unique(cqi_cpos2id("DICKENS.pos", cpos))
cqi_id2freq("DICKENS.pos", pids)

wids <- unique(cqi_cpos2id("DICKENS.word", cpos))
cqi_id2freq("DICKENS.word", wids)

lids <- unique(cqi_cpos2id("DICKENS.lemma", cpos))
cqi_id2freq("DICKENS.lemma", lids)

## End(Not run)
```

cqi_list_corpora	<i>List of available corpora</i>
------------------	----------------------------------

Description

Obtain the names of the installed corpora.

Usage

```
cqi_list_corpora()
```

Details

The available corpora are referenced in the registry (which is a directory located by default in ‘/usr/local/share/cwb’ on Unix-like systems). See [cqp_registry](#) to learn how to set the *registry* directory.

The name of a corpus is always written in uppercase letters (with possibly underscores, dashes and digits).

Value

cqi_list_corpora returns a list of the available corpora as specified in the registry.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqp_registry](#), [cqi_full_name](#), [cqi_list_subcorpora](#), [cqi_query](#), [cqi_attributes](#), [subcorpus](#), [corpus](#).

Examples

```
## Not run:
cqi_list_corpora()

## End(Not run)
```

cqi_list_subcorpora	<i>List of currently defined subcorpora</i>
---------------------	---------------------------------------------

Description

Get a list of the currently defined subcorpora in a given corpus.

Usage

```
cqi_list_subcorpora(corpus)
```

Arguments

corpus (*string*) the name of the parent corpus (in uppercase letters).

Details

This function returns a list of all the named subcorpora currently defined. A subcorpus is the result of a query and its name is typically declared in the second argument of the [cqi_query](#) function.

Value

cqi_list_subcorpora returns a list of the existing subcorpora in the specified corpus.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

See Also

[cqi_list_corpora](#), [cqi_query](#), [cqi_subcorpus_size](#), [cqi_attributes](#), [cqi_fdist](#), [subcorpus](#), [corpus](#).

Examples

```
## Not run:
cqi_query("DICKENS", "Foo", "\"interesting\"")
# The returned string should contain at least "Last" and "Foo"
cqi_list_subcorpora("DICKENS")

## End(Not run)
```

cqi_query	<i>Query an indexed corpus</i>
-----------	--------------------------------

Description

The `cqi_query` function executes a query on the specified corpus. The query syntax is described in the *CQP Query Language Tutorial* (see below).

Usage

```
cqi_query(mother, child, query)
```

Arguments

mother	(<i>string</i>) the name of the corpus.
child	(<i>string</i>) the name given to the subcorpus which will contain the results of the query.
query	(<i>string</i>) the query string.

Details

The query string must be properly quoted in order to be syntactically correct when it is passed to the CQP interpreter.

The name of a corpus is always written in uppercase letters. The name of a subcorpus must start with an uppercase letter and can contain both uppercase and lowercase letters, underscores, dashes and digits.

Value

`cqi_query` does not return anything. In order to manipulate the results, one must invoke the `cqi_dump_subcorpus` function.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

- *CQP Query Language Tutorial*: http://cwb.sourceforge.net/files/CQP_Tutorial.pdf
- *Corpus Encoding Tutorial*: http://cwb.sourceforge.net/files/CWB_Encoding_Tutorial.pdf

See Also

[cqi_dump_subcorpus](#), [cqi_drop_subcorpus](#), [cqi_attributes](#), [cqi_attribute_size](#), [subcorpus](#), [corpus](#).

Conversion functions: [cqi_alg2cpos](#), [cqi_cpos2alg](#), [cqi_cpos2id](#), [cqi_cpos2lbound](#), [cqi_cpos2rbound](#), [cqi_cpos2str](#), [cqi_cpos2struc](#), [cqi_id2cpos](#), [cqi_id2freq](#), [cqi_id2str](#), [cqi_regex2id](#), [cqi_str2id](#), [cqi_struc2cpos](#), [cqi_struc2str](#).

Examples

```
## Not run:
cqi_query("DICKENS","Foo","\\"eas(y|ier|iest)\\","")
cqi_dump_subcorpus("DICKENS:Foo",0,10)
cqi_drop_subcorpus("DICKENS:Foo")

cqi_query("DICKENS","Go","[lemma=\\\"go\\\"] \\\\\"and\\\" [];")
cqi_dump_subcorpus("DICKENS:Go",0,10)
cqi_drop_subcorpus("DICKENS:Go")

cqi_query("DICKENS","NP","[pos=\\\"DT\\\"] @[pos=\\\"JJ\\\"]? [pos=\\\"NNS?\\\"]");
cqi_dump_subcorpus("DICKENS:NP",0,10)
cqi_drop_subcorpus("DICKENS:NP")

## End(Not run)
```

cqi_regex2id

Find IDs by regular expression

Description

Build a list of IDs of positional attributes whose names match a regular expression.

Usage

```
cqi_regex2id(attribute, regex)
```

Arguments

attribute	(string) the name of a positional attribute.
regex	(string) a regular expression.

Details

This function looks in the index files corresponding to the specified positional attribute, finds the strings described by the regular expression and returns the associated indices.

Value

cqi_regex2id returns a list of integers.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_id2cpos](#), [cqi_id2freq](#), [cqi_id2str](#), [cqi_str2id](#), [cqi_struc2cpos](#), [cqi_struc2str](#).

Examples

```
## Not run:
rgx <- "eas(y|ier|iest)"
wids <- cqi_regex2id("DICKENS.word", rgx)
cqi_id2str("DICKENS.word", wids)

rgx <- "V.*"
pids <- cqi_regex2id("DICKENS.pos", rgx)
cqi_id2str("DICKENS.pos", pids)

rgx <- "V[aeiou].+e"
lids <- cqi_regex2id("DICKENS.lemma", rgx)
cqi_id2str("DICKENS.lemma", lids)

## End(Not run)
```

cqi_struc2cpos

Convert structural indices

Description

Get the corpus positions or the string associated to a structural position.

Usage

```
cqi_struc2cpos(attribute, struc)
cqi_struc2str(attribute, ids)
```

Arguments

attribute	(<i>string</i>) the qualified name of an attribute.
struc	(<i>integer</i>)
ids	(<i>integer vector</i>)

Details

The structural attributes correspond to XML pairs of tags in the corpus. Matching pairs of XML start and end tags are encoded as token regions. The *struc* parameter is the index of a particular pair of tags. The `cqi_struc2cpos` function associates the corpus positions of the specified pair.

Value

The `cqi_struc2cpos` function returns a pair of corpus positions: the corpus positions of the first token (immediately following the start tag) and of the last token (immediately preceding the end tag) of the token region.

The `cqi_struc2str` function returns a string.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_alg2cpos](#), [cqi_cpos2alg](#), [cqi_cpos2id](#), [cqi_cpos2lbound](#), [cqi_cpos2rbound](#), [cqi_cpos2str](#), [cqi_cpos2struc](#), [cqi_id2cpos](#), [cqi_id2freq](#), [cqi_id2str](#), [cqi_regex2id](#), [cqi_str2id](#).

Examples

```
## Not run:
for (i in 0:10) {
  print(cqi_struc2cpos("DICKENS.novel", i))
}

cqi_struc2str("DICKENS.novel_title", 1:10)
cqi_struc2str("DICKENS.chapter_title", 1:10)

## End(Not run)
```

cqp_flist	Create a frequency list
-----------	-------------------------

Description

Create a frequency list on any phenomenon available in a CWB corpus: frequency of forms of a positional attribute, frequency of a value of a structural attribute, or frequency of forms found in an anchor (*match*, *matchend*, *target*, *keyword*) of a subcorpus.

Usage

```
cqp_flist(x, ...)
```

```
## S3 method for class 'cqp_attr'
```

```
cqp_flist(x, cutoff = 0, ...)
```

```
## S3 method for class 'cqp_corpus'
```

```
cqp_flist(x, attribute, cutoff = 0, ...)
```

```
## S3 method for class 'cqp_subcorpus'
```

```
cqp_flist(x, anchor, attribute, left.context = 0,
          right.context = 0, cutoff = 0, offset = 0, ...)
```

Arguments

x	An attribute, a corpus or a subcorpus object, created with the function corpus or subcorpus .
attribute	The attribute giving the modality to be counted. May be a structural attribute if x is a corpus.
anchor	Where to count the modality : on the <i>match</i> , <i>matchend</i> , <i>target</i> or <i>keyword</i> anchor of a subcorpus.
left.context	Consider also <i>n</i> tokens on the left of the selected anchor for counting frequencies.
right.context	Consider also <i>n</i> tokens on the right of the selected anchor for counting frequencies.
offset	Switch the anchor at <i>n</i> tokens from the left (if negative) or to the right of the anchor given above.
cutoff	Keep only frequencies above this cutoff. If cutoff is 0, keep all forms. If cutoff is between 0 and 1 (exclusive), the (cutoff * 100) % more frequent forms are kept.
...	Ignored.

Value

Returns a named numeric vector.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
 Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqp_fable](#), [cqp_kwic](#), [subcorpus](#), [corpus](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
fl <- cqp_flist(c, "word");

sc <- subcorpus(c, "interesting" "to" @ []);
fl <- cqp_flist(sc, "target", "word");

sc <- subcorpus(c, "interesting");
fl <- cqp_flist(sc, "target", "lemma", left.context = 10, right.context = 10, cutoff = 5)

## End(Not run)
```

cqp_fable

Create a frequency table

Description

Create a frequency table either with a corpus or with a subcorpus. With a corpus, a frequency table is based on two attributes (structural or positional). With a subcorpus object, a frequency table is based on two anchors (*match*, *matchend*, *target*, *keyword*) and a positional attribute for each anchor.

Usage

```
cqp_fable(x, ...)

## S3 method for class 'cqp_corpus'
cqp_fable(x, attribute1, attribute2, attribute1.use.id = FALSE,
          attribute2.use.id = FALSE, structural.attribute.unique.id = FALSE,
          subcorpus = NULL, ...)

## S3 method for class 'cqp_subcorpus'
cqp_fable(x, anchor1, attribute1,
          anchor2, attribute2, cutoff = 0, ...)
```

Arguments

x	An rcqp object, created with corpus or subcorpus .
attribute1	The attribute for the modalities of the first variable of the cross-tabulation. If x is a subcorpus, positional attribute only.
attribute2	The attribute for the modalities of the second variable of the cross-tabulation. If x is a subcorpus, positional attribute only.

attribute1.use.id	If attribute1 is a structural attribute and has values (see cqi_structural_attribute_has_values), switch between region ids (struc) and values (default).
attribute2.use.id	If attribute2 is a structural attribute and has values (see cqi_structural_attribute_has_values), switch between region ids (struc) and values (default).
structural.attribute.unique.id	Count tokens or ids. See details for more info.
subcorpus	Not implemented yet.
anchor1	The anchor for individuals of the first variable, if x is a subcorpus (anchor may be : <i>match</i> , <i>matchend</i> , <i>target</i> , <i>keyword</i>).
anchor2	The anchor for individuals of the second variable, if x is a subcorpus (anchor may be : <i>match</i> , <i>matchend</i> , <i>target</i> , <i>keyword</i>).
cutoff	Filter the frequency table.
...	Ignored.

Details

Some explanations for the `structural.attribute.unique.id` option (see the vignette *RcqpIntroduction*).

Positional attributes (and structural attributes having values) are represented with their string values rather than with ids. For positional attributes, it is only a matter of presentation, since each id has its own string; but for structural attributes having values, it may entail a different counting, since these values are not unique: occurrences of phenomena belonging to different structs are then counted together if two structs have the same value. You can force the use of ids rather than string values with the `attribute1.use.id` and `attribute2.use.id` options.

Counts are made on token basis, i.e. each token of the corpus is an individual on which the two modalities (attributes) are considered. If you use two structural attributes as arguments in `cqp_ftable`, and one of them does not have values, then the third column counts the number of tokens. In the following example, each line gives the length (in number of tokens, third column) of each sentence (second column) in each novel represented by its title:

```
c <- corpus("DICKENS");
f <- cqp_ftable(c, "novel_title", "s")
f[1:10,]
```

If both structural attributes have values, you may want to count the number of times the modalities are cooccurring, rather than the total number of tokens included in these cooccurrences. For that purpose, you can use the `structural.attribute.unique.id=TRUE` option. In the following example, we count the number of time each head appears in each novel :

```
f <- cqp_ftable(c, "novel_title", "pp_h", structural.attribute.unique.id=TRUE)
f[1:10,]
```

Here on the contrary, we count the total number of tokens in each prepositional phrase having a given head :

```
f <- cqp_ftable(c, "novel_title", "pp_h")
f[1:10,]
```

Value

A frequency table stored as a flat (3-column) dataframe : for each observed combination of modalities, the first column gives the modality in the first variable, the second column the modality in the second variable, and the third column the observed frequency of the cooccurrence.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqp_flist](#), [cqp_kwic](#), [subcorpus](#).

cqp_kwic

Create a "keyword in context" list with a subcorpus.

Description

Create a "keyword in context" (aka *kwic*) list with a subcorpus, with parameters for arranging and filtering items. The list can then be printed chunk by chunk.

Usage

```
cqp_kwic(x, ...)
```

```
## S3 method for class 'cqp_subcorpus'
cqp_kwic(x, right.context = 20, left.context = 20, ...)
```

Arguments

x	An object created with subcorpus .
right.context	The number of characters on the right.
left.context	The number of characters on the left.
...	Ignored.

Value

Create a `cqp_kwic` object ready to be printed (with more options for the display) with `print` (see [print.cqp_kwic](#)).

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[corpus](#), [subcorpus](#), [print.cqp_kwic](#) [write.cqp_corpus](#).

cqp_registry	<i>The CQP registry</i>
--------------	-------------------------

Description

Setting the CQP registry for rcqp.

Usage

```
cqi_getRegistry()
cqi_setRegistry(path)
```

Arguments

`path` *(string)* the full path of the registry directory.

Details

Corpora are declared in files located in a directory called *registry* (see CWB documentation for more details). One of the mechanisms provided by CWB in order to set the path of the *registry* directory is to define the environment variable `CORPUS_REGISTRY`.

In rcqp, the `CORPUS_REGISTRY` environment variable is first read. If it is not set, rcqp tries to use the default registry directory, which varies according to your system (on most Unixes, it is `‘/usr/local/share/cwb/registry’`; on Windows, it is `‘C:\CWB\Registry’`). If this default directory does not exist, rcqp cannot proceed further. At this point, you can use the `cqi_setRegistry` function to specify the path of the registry directory.

The path passed to the `cqi_setRegistry` function must exist. Note that, once it is set, the path of the registry may not be changed.

In order to set the `CORPUS_REGISTRY` variable, you can

- either use the mechanisms provided by your system to set the environment variable before the R command is executed.
- or define the variable from the R console using the [Sys.setenv](#) function. This must be done before the package is loaded.
- or define this variable in the configuration file `‘~/Renviron’` which is read by R at startup. In that case, the environment variable is defined only for rcqp, and will not be seen by any other CWB program.

Value

`cqi_getRegistry` returns a vector with the path of the registry directory if it has been set, otherwise the NULL string. `cqi_setRegistry` does not return anything.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
 Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

Source

The IMS Open Corpus Workbench (CWB) at <http://cwb.sourceforge.net/>

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqi_list_corpora](#), [cqi_list_subcorpora](#), [cqi_query](#).

Examples

```
## Not run:
Sys.setenv(CORPUS_REGISTRY="/users/vhugo/cwb/registry")
cqi_setRegistry("/users/vhugo/cwb/registry")

## End(Not run)
```

 Extraction

Extracting/addressing an attribute from a corpus.

Description

Create a 'cqp_attr' S3 object, holding a cqp attribute (structural or positional).

Usage

```
## S3 method for class 'cqp_corpus'
i[[j, ...]]
## S3 method for class 'cqp_corpus'
x$name
```

Arguments

i	The name of an attribute
j	Ignored
x	an rcq_corpus object
name	an attribute name
...	Ignored

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
 Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

print.cqp_flist	<i>Print a CWB object</i>
-----------------	---------------------------

Description

Print a CWB object in an exhaustive way. A `cqp_flist` object is printed just as an integer vector; a corpus is printed as a dataframe where rows represent tokens and columns represent attributes (positional as well as structural); a subcorpus is printed as a `cqp_kwic` list (or concordance); a `cqp_kwic` object is printed... as expected.

Usage

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

## S3 method for class 'cqp_corpus'
print(x, from = 0, to = 20, use_value = TRUE, ...)

## S3 method for class 'cqp_subcorpus'
print(x, positional.attribute = "word", from = 0, to = 10, ...)

## S3 method for class 'cqp_kwic'
print(x, from = 0, to = min(20, nrow(x)-1),
      print_tokens = function(x, cpos) cqi_cpos2str(
        paste(attr(x, "parent.cqp_corpus.name"), "word", sep = "."), cpos),
      left.separator = " <<", right.separator = ">> ", ...)

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

Arguments

<code>x</code>	An rcqp object, created with <code>cqp_flist</code> , <code>corpus</code> , <code>subcorpus</code> , <code>cqp_kwic</code> , or <code>\$.cqp_corpus</code> .
<code>from</code>	Select first tokens (for a corpus object) or matches (for a subcorpus or <code>cqp_kwic</code> object) to be printed. As for all CWB data, this index is 0-based.
<code>to</code>	Select last tokens (for a corpus object) or matches (for a subcorpus or <code>cqp_kwic</code> object) to be printed. As for all CWB data, this index is 0-based.
<code>use_value</code>	use value or id for a structural attribute with values.
<code>positional.attribute</code>	Print each token of the subcorpus using the given positional attribute.
<code>print_tokens</code>	A function allowing a fine access to the way tokens are displayed in a concordance.
<code>left.separator</code>	String inserted at the left of the keyword in a concordance.
<code>right.separator</code>	String inserted at the right of the keyword in a concordance.
<code>...</code>	Ignored.

Details

The print command applied on a subcorpus object is a shortcut for :

```
k <- cqp_kwic(subcorpus)
print(k)
```

Creating a cqp_kwic object gives more control on the printing options.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[corpus](#), [subcorpus](#), [cqp_kwic](#), [cqp_flist](#), [cqp_ftable](#), [summary.cqp_flist](#), [size.cqp_corpus](#), [size.cqp_subcorpus](#), [summary.cqp_corpus](#), [summary.cqp_subcorpus](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
c;

sc <- subcorpus(c, "interesting");
sc;

k <- cqp_kwic(sc);
k;

fl <- cqp_flist(c, "word");
fl;

## End(Not run)
```

rcqp

~ Corpus Query Protocol ~

Description

```
Package: rcqp
Type: Package
Version: 0.5
Date: 2018-03-07
License: GPL-2 | file LICENCE
```

Details

rcqp is an implementation in R of the Corpus Query Protocol.

See the [cqp_registry](#) help page or type `?cqp_registry` from the R console to see how to set the "registry" directory.

Author

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.

Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

- *CQP Query Language Tutorial*: http://cwb.sourceforge.net/files/CQP_Tutorial.pdf
- *Corpus Encoding Tutorial*: http://cwb.sourceforge.net/files/CWB_Encoding_Tutorial.pdf

See Also

List of available help pages:

[cqi_attribute_size](#)
[cqi_attributes](#)
[cqi_cpos2id](#)
[cqi_dump_subcorpus](#)
[cqi_fdist](#)
[cqi_full_name](#)
[cqi_id2cpos](#)
[cqi_list_corpora](#)
[cqi_list_subcorpora](#)
[cqi_query](#)
[cqi_regex2id](#)
[cqi_struct2cpos](#)
[cqp_registry](#)
[cqi_corpus_info](#)

Object model:

[corpus](#)
[subcorpus](#)
[cqp_flist](#)
[cqp_ftable](#)
[cqp_kwic](#)

region_sizes	<i>Size of regions of a structural attribute.</i>
--------------	---------------------------------------------------

Description

Compute the size (in number of tokens) of the regions of a structural attribute.

Usage

```
region_sizes(attribute)
## S3 method for class 'cqp_attr'
region_sizes(attribute)
```

Arguments

attribute A cqp_attr (type: structural attribute) (created for instance with [\\$.cqp_corpus](#)).

Value

An integer vector: for each region of this attribute, the number of tokens belonging to that region.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqp_flist](#), [cqp_ftable](#), [cqp_kwic](#), [corpus](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
s <- region_sizes(c$pos);

## End(Not run)
```

size	<i>Size of a corpus or subcorpus</i>
------	--------------------------------------

Description

Size of a corpus (number of tokens) or size of a subcorpus (number of matches).

Usage

```
size(x)
```

```
## S3 method for class 'cqp_corpus'
size(x)
```

```
## S3 method for class 'cqp_subcorpus'
size(x)
```

Arguments

x A cqp_corpus object (created with the function [corpus](#)) or a cqp_subcorpus object (created with the function [subcorpus](#))

Value

An integer vector of length 1: the number of tokens if x is a corpus or the number of matches if it is a subcorpus.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[corpus](#), [subcorpus](#), [summary.cqp_corpus](#), [summary.cqp_subcorpus](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
size(c);

sc <- subcorpus(c, "interesting");
size(sc);

## End(Not run)
```

sort.cqp_kwic	<i>Sort a kwic list</i>
---------------	-------------------------

Description

Sort a kwic list with respect to an anchor and a positional attribute.

Usage

```
## S3 method for class 'cqp_kwic'
sort(x, decreasing = FALSE, sort.anchor = "match",
     sort.attribute = "word", sort.offset = 0, ...)
```

Arguments

x	A cqp_kwic object.
decreasing	Reverse order.
sort.anchor	The anchor (<i>match</i> , <i>matchend</i> , <i>target</i> , <i>keyword</i>) to be used as sorting key.
sort.attribute	The attribute (such as <i>word</i> , <i>lemma</i> ... according to the available attributes in a given corpus) for sorting the lines of the concordance.
sort.offset	Sort on the selected anchor, or <i>n</i> tokens before or after it.
...	Ignored.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
 Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[cqp_kwic](#), [print.cqp_kwic](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
sc <- subcorpus("DICKENS", "interesting");
kwic <- cqp_kwic <- (sc);
kwic <- sort(kwic)
print(kwic);

## End(Not run)
```

subcorpus	Create a subcorpus.
-----------	---------------------

Description

Create a subcorpus held in the CWB library.

Usage

```
subcorpus(corpus, query)
```

Arguments

corpus	An rcqp object created with corpus .
query	A CQP query.

Details

The *subcorpus* function creates internally a subcorpus name for the CWB library.

Value

A subcorpus object.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[corpus](#), [cqp_flist](#), [print.cqp_subcorpus](#), [summary.cqp_subcorpus](#), [cqp_ftable](#), [cqp_kwic](#).

Examples

```
## Not run:  
c <- corpus("DICKENS");  
sc <- subcorpus("DICKENS", "interesting");  
  
## End(Not run)
```

summary.cqp_flist	<i>Print summary for CWB objects.</i>
-------------------	---------------------------------------

Description

Print a summary for a corpus, a subcorpus, or a frequency list.

Usage

```
## S3 method for class 'cqp_flist'
summary(object, ...)

## S3 method for class 'cqp_corpus'
summary(object, ...)

## S3 method for class 'cqp_subcorpus'
summary(object, ...)

## S3 method for class 'cqp_attr'
summary(object, ...)
```

Arguments

object	An rcqp object, created with cqp_flist , corpus , subcorpus , or \$.cqp_corpus .
...	Ignored.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
 Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[corpus](#), [subcorpus](#), [write.cqp_corpus](#), [size.cqp_corpus](#), [size.cqp_subcorpus](#), [cqp_kwic](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
c;

sc <- subcorpus(c, "interesting");
sc;

k <- cqp_kwic(sc);
k;
```



```
fl <- cqp_flist(c, "word");
fl;

## End(Not run)
```

Type and token	<i>Print values of cqp attribute</i>
----------------	--------------------------------------

Description

Get vectors of tokens or types for a cqp attribute.

Usage

```
ntype(attribute, ...)
types(attribute, ...)
nregion(attribute, ...)
regions(attribute, ...)
ntoken(attribute, ...)
tokens(attribute, ...)

## S3 method for class 'cqp_attr'
ntype(attribute, ...)
## S3 method for class 'cqp_attr'
types(attribute, ...)
## S3 method for class 'cqp_attr'
nregion(attribute, ...)
## S3 method for class 'cqp_attr'
regions(attribute, ...)
## S3 method for class 'cqp_attr'
ntoken(attribute, ...)
## S3 method for class 'cqp_attr'
tokens(attribute, ...)
```

Arguments

attribute	A cqp_attr object, created with \$.cqp_corpus .
...	Ignored.

Value

For positional attributes :

- ntoken is the number of tokens (== size(corpus))

- tokens is a character vector with the actual list of tokens
- ntype is the number of different values
- types is a character vector with the actual different values

For structural attributes :

- nregion is the actual number of regions (a numeric vector of length 1)
- tokens the region id ("struc") of each token (a numeric vector)
- regions (only if the structural attribute "has values") : the values found on each region (a character vector)

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.
Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[\\$.cqp_corpus](#).

Examples

```
## Not run:
c <- corpus("DICKENS");
pos <- c$pos
types(pos);
ntokens(types(pos));

## End(Not run)
```

write.cqp_corpus

Save a CWB corpus in a file in tabular representation

Description

Save a CWB corpus in a file as a dataframe where each column represents an attribute (positional or structural).

Usage

```
## S3 method for class 'cqp_corpus'
write(corpus, filename, from = 0, to = 1000, ...)
```

Arguments

corpus	An object created with corpus .
filename	The file to be used.
from	First token to save (token <i>cpos</i> of the first line of the dataframe).
to	Last token to save (token <i>cpos</i> of the last line of the dataframe).
...	Not used.

Author(s)

Bernard Desgraupes - <bernard.desgraupes@u-paris10.fr> - University Paris-10.

Sylvain Loiseau - <sylvain.loiseau@univ-paris13.fr> - University Paris-13.

References

<http://cwb.sourceforge.net/documentation.php>

See Also

[corpus](#), [print.cqp_corpus](#), [summary.cqp_corpus](#).

Index

- *Topic **attribute**
 - cqi_attribute_size, 4
- *Topic **corpus positions**
 - cqi_cpos2id, 6
- *Topic **corpus position**
 - cqi_id2cpos, 11
 - cqi_regex2id, 16
 - cqi_struct2cpos, 17
- *Topic **corpus, attributes**
 - cqi_attributes, 3
- *Topic **corpus, query**
 - cqi_query, 15
- *Topic **corpus, registry**
 - cqi_list_corpora, 13
- *Topic **corpus, subcorpus**
 - cqi_list_subcorpora, 14
- *Topic **corpus**
 - cqi_full_name, 10
- *Topic **cqp_attr**
 - Extraction, 24
- *Topic **frequency**
 - cqi_fdist, 9
- *Topic **package**
 - rcqp, 26
- *Topic **registry**
 - cqp_registry, 23
- *Topic **subcorpus, indices**
 - cqi_dump_subcorpus, 8
- [[.cqp_corpus (Extraction), 24
- \$.cqp_corpus, 25, 28, 32–34
- \$.cqp_corpus (Extraction), 24
- corpus, 2, 14–16, 19, 20, 23, 25–29, 31, 32, 35
- cqi_alg2cpos, 12, 16, 18
- cqi_alg2cpos (cqi_cpos2id), 6
- cqi_attribute_size, 3, 4, 16, 27
- cqi_attributes, 3, 5, 14–16, 27
- cqi_corpus_info, 5, 27
- cqi_cpos2alg, 12, 16, 18
- cqi_cpos2alg (cqi_cpos2id), 6
- cqi_cpos2id, 6, 12, 16, 18, 27
- cqi_cpos2lbound, 12, 16, 18
- cqi_cpos2lbound (cqi_cpos2id), 6
- cqi_cpos2rbound, 12, 16, 18
- cqi_cpos2rbound (cqi_cpos2id), 6
- cqi_cpos2str, 12, 16, 18
- cqi_cpos2str (cqi_cpos2id), 6
- cqi_cpos2struc, 12, 16, 18
- cqi_cpos2struc (cqi_cpos2id), 6
- cqi_drop_subcorpus, 16
- cqi_drop_subcorpus
 - (cqi_dump_subcorpus), 8
- cqi_dump_subcorpus, 8, 10, 16, 27
- cqi_fdist, 9, 15, 27
- cqi_fdist1 (cqi_fdist), 9
- cqi_fdist2 (cqi_fdist), 9
- cqi_full_name, 10, 14, 27
- cqi_getRegistry (cqp_registry), 23
- cqi_id2cpos, 7, 11, 16–18, 27
- cqi_id2freq, 7, 16–18
- cqi_id2freq (cqi_id2cpos), 11
- cqi_id2str, 7, 16–18
- cqi_id2str (cqi_id2cpos), 11
- cqi_lexicon_size (cqi_attribute_size), 4
- cqi_list_corpora, 8, 11, 13, 15, 24, 27
- cqi_list_subcorpora, 8, 10, 14, 14, 24, 27
- cqi_query, 8, 14, 15, 15, 24, 27
- cqi_regex2id, 7, 12, 16, 16, 18, 27
- cqi_setRegistry (cqp_registry), 23
- cqi_str2id, 7, 12, 16–18
- cqi_str2id (cqi_id2cpos), 11
- cqi_struct2cpos, 7, 12, 16, 17, 17, 27
- cqi_struct2str, 4, 5, 7, 12, 16, 17
- cqi_struct2str (cqi_struct2cpos), 17
- cqi_structural_attribute_has_values, 3, 21
- cqi_structural_attribute_has_values
 - (cqi_attribute_size), 4
- cqi_subcorpus_size, 15
- cqi_subcorpus_size
 - (cqi_dump_subcorpus), 8
- cqp_flist, 19, 22, 25–28, 31, 32
- cqp_ftable, 20, 20, 26–28, 31
- cqp_kwic, 2, 20, 22, 22, 25–28, 30–32
- cqp_registry, 13, 14, 23, 27
- Extraction, 24

nregion (Type and token), 33
ntoken (Type and token), 33
ntype (Type and token), 33

print.cqp_attr (print.cqp_flist), 25
print.cqp_corpus, 2, 35
print.cqp_corpus (print.cqp_flist), 25
print.cqp_flist, 25
print.cqp_kwic, 22, 23, 30
print.cqp_kwic (print.cqp_flist), 25
print.cqp_subcorpus, 31
print.cqp_subcorpus (print.cqp_flist), 25

rcqp, 26
rcqp-package (rcqp), 26
region_sizes, 28
regions (Type and token), 33

size, 29
size.cqp_corpus, 26, 32
size.cqp_subcorpus, 26, 32
sort.cqp_kwic, 30
subcorpus, 2, 14–16, 19, 20, 22, 23, 25–27,
29, 31, 32
summary.cqp_attr (summary.cqp_flist), 32
summary.cqp_corpus, 2, 26, 29, 35
summary.cqp_corpus (summary.cqp_flist),
32
summary.cqp_flist, 26, 32
summary.cqp_subcorpus, 26, 29, 31
summary.cqp_subcorpus
(summary.cqp_flist), 32
Sys.setenv, 23

tokens (Type and token), 33
Type and token, 33
types (Type and token), 33

write.cqp_corpus, 2, 23, 32, 34