Package ‘nzilbb.labbcat’

July 7, 2021

Version 1.0-1
Date 2021-07-07
Title Accessing Data Stored in 'LaBB-CAT' Instances
Imports jsonlite, httr, stringr, utils, rstudioapi
Description 'LaBB-CAT' is a web-based language corpus management
   system developed by the New Zealand Institute of Language, Brain
   and Behaviour (NZILBB) - see <https://labbcat.canterbury.ac.nz>. This package defines functions for accessing corpus data in a 'LaBB-CAT' instance. You must have at least version 20210601.1528 of 'LaBB-CAT' to use this package.
License GPL (>= 3)
Copyright New Zealand Institute of Language, Brain and Behaviour,
   University of Canterbury
URL https://nzilbb.github.io/labbcat-R/,
   https://labbcat.canterbury.ac.nz
RoxygenNote 7.1.1
Suggests testthat (>= 2.1.0)
NeedsCompilation no
Author Robert Fromont [aut, cre] (<https://orcid.org/0000-0001-5271-5487>)
Maintainer Robert Fromont <robert.fromont@canterbury.ac.nz>
Repository CRAN
Date/Publication 2021-07-07 14:50:02 UTC

R topics documented:

  countAnnotations .......................................................... 3
deleteTranscript .......................................................... 4
getAllUtterances .......................................................... 4
getAnchors ................................................................. 6
getAnnotations ............................................................. 7
getAvailableMedia ........................................................ 9
getCorpusIds .............................................................. 10
getDeserializerDescriptors ............................................. 10
getDictionaries ........................................................... 11
getDictionaryEntries ..................................................... 12
getFragments .............................................................. 13
getGraphIds ............................................................... 14
getGraphIdsInCorpus ..................................................... 15
getGraphIdsWithParticipant ............................................ 16
getId ...................................................................... 16
getLayer ................................................................. 17
getLayerIds ............................................................... 18
getLayers ................................................................. 19
getMatchAlignments ..................................................... 20
getMatches ............................................................... 21
getMatchingGraphIds ................................................... 25
getMatchingParticipantIds ............................................. 27
getMatchingTranscriptIds ............................................... 28
getMatchLabels .......................................................... 30
getMedia ................................................................. 31
getMediaTracks .......................................................... 32
getParticipantAttributes ................................................. 33
getParticipantIds ........................................................ 34
getSerializerDescriptors ............................................... 34
getSoundFragments ...................................................... 35
getSystemAttribute ...................................................... 37
getTranscriptAttributes ................................................ 37
getTranscriptIds ........................................................ 38
getTranscriptIdsInCorpus ............................................... 39
getTranscriptIdsWithParticipant ..................................... 40
getUserInfo ............................................................. 40
labbcatCredentials ....................................................... 41
labbcatTimeout .......................................................... 42
newTranscript ........................................................... 43
nzilbh.labbcat ............................................................ 44
praatScriptCentreOfGravity ............................................. 47
praatScriptFastTrack .................................................... 48
praatScriptFormants ...................................................... 51
praatScriptIntensity ...................................................... 53
praatScriptPitch ........................................................ 54
processWithPraat ........................................................ 56
updateTranscript ......................................................... 59
countAnnotations

**Description**

Returns the number of annotations on the given layer of the given transcript.

**Usage**

```r
countAnnotations(labbcat.url, id, layer.id)
```

**Arguments**

- `labbcat.url` URL to the LaBB-CAT instance
- `id` A transcript ID (i.e. transcript name)
- `layer.id` A layer ID

**Value**

The number of annotations on that layer

**See Also**

`getTranscriptIds`, `getTranscriptIdsInCorpus`, `getTranscriptIdsWithParticipant`

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Count the number of words in UC427_ViktoriaPapp_A_ENG.eaf
token.count <- countAnnotations(labbcat.url, "UC427_ViktoriaPapp_A_ENG.eaf", "orthography")

## End(Not run)
```
deleteTranscript  

Delete a transcript from the corpus.

Description

This function deletes the given transcript, and all associated files.

Usage

deleteTranscript(labbcat.url, id)

Arguments

labbcat.url  
URL to the LaBB-CAT instance

id  
The ID transcript to delete.

Details

For this function to work, the credentials used to connect to the server must have at least 'edit' access.

Value

The ID of the deleted transcript

Examples

## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

delteTranscript(labbcat.url, "my-transcript.eaf")

## End(Not run)

getAllUtterances  

Get all utterances of participants.

Description

Identifies all utterances of a given set of participants.
getAllUtterances

Usage

ggetAllUtterances(
  labbcat.url,
  participant.ids,
  transcript.types = NULL,
  main.participant = TRUE,
  max.matches = NULL,
  no.progress = FALSE
)

Arguments

labbcat.url URL to the LaBB-CAT instance
participant.ids A list of participant IDs to identify the utterances of.
transcript.types An optional list of transcript types to limit the results to. If null, all transcript
types will be searched.
main.participant TRUE to search only main-participant utterances, FALSE to search all ut-
terances.
max.matches The maximum number of matches to return, or null to return all.
no.progress TRUE to supress visual progress bar. Otherwise, progress bar will be shown
when interactive().

Value

A data frame identifying matches, containing the following columns:

• SearchName A name based on the pattern – the same for all rows
• Number Row number
• Transcript Name of the transcript in which the match was found
• Line The start offset of the utterance/line
• LineEnd The end offset of the utterance/line
• MatchId A unique ID for the matching target token
• Before.Match Transcript text immediately before the match
• Text Transcript text of the match
• Before.Match Transcript text immediately after the match
• Target.word Text of the target word token
• Target.word.start Start offset of the target word token
• Target.word.end End offset of the target word token
• Target.segment Label of the target segment (only present if the segment layer is included in
  the pattern)
getAnchors

• Target.segment.start Start offset of the target segment (only present if the segment layer is included in the pattern)
• Target.segment.end End offset of the target segment (only present if the segment layer is included in the pattern)

See Also
getParticipantIds

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## get all utterances of the given participants
participant.ids <- getParticipantIds(labbcat.url)[1:3]
results <- getAllUtterances(labbcat.url, participant.ids)

## results$MatchId can be used to access results
## End(Not run)
```

getAnchors

*Gets the given anchors in the given transcript.*

Description

Lists the given anchors in the given transcript.

Usage

`getAnchors(labbcat.url, id, anchor.id, page.length = 1000)`

Arguments

- `labbcat.url` URL to the LaBB-CAT instance
- `id` A transcript ID (i.e. transcript name)
- `anchor.id` A vector of anchor IDs (or a string representing one anchor ID)
- `page.length` In order to prevent timeouts when there are a large number of matches or the network connection is slow, rather than retrieving anchors in one big request, they are retrieved using many smaller requests. This parameter controls the number of anchors retrieved per request.
getAnnotations

Value
A named list of anchors, with members:

• `id` The annotation’s unique ID,
• `offset` The offset from the beginning (in seconds if it’s a transcript of a recording, or in characters if it’s a text document)
• `confidence` A rating from 0-100 of the confidence of the offset, e.g. 10: default value, 50: force-aligned, 100: manually aligned

See Also
getAnnotations

Examples
```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get the first 20 orthography tokens in UC427_ViktoriaPapp_A_ENG.eaf
orthography <- getAnnotations(labbcat.url, "UC427_ViktoriaPapp_A_ENG.eaf", "orthography", 20, 0)

## Get the start anchors for the above tokens
word.starts <- getAnchors(labbcat.url, "UC427_ViktoriaPapp_A_ENG.eaf", orthography$startId)

## End(Not run)
```

getAnnotations

- **Gets the annotations on the given layer of the given transcript.**

Description
Returns the annotations on the given layer of the given transcript.

Usage
```
getAnnotations(
    labbcat.url,
    id,
    layer.id,
    page.length = NULL,
    page.number = NULL
)
```
getAnnotations

Arguments

- `labbcat.url`: URL to the LaBB-CAT instance
- `id`: A transcript ID (i.e. transcript name)
- `layer.id`: A layer ID
- `page.length`: The maximum number of annotations to return, or null to return all
- `page.number`: The zero-based page number to return, or null to return the first page

Value

A named list of annotations, with members:

- `id`: The annotation’s unique ID
- `layerId`: The name of the layer it comes from
- `label`: The value of the annotation
- `startId`: The ID of the start anchor,
- `endId`: The ID of the end anchor,
- `parentId`: The ID of the parent annotation,
- `ordinal`: The ordinal of the annotation among its peers,
- `confidence`: A rating from 0-100 of the confidence of the label e.g. 10: default value, 50: automatically generated, 100: manually annotated

See Also

- `getTranscriptIds`
- `getTranscriptIdsInCorpus`
- `getTranscriptIdsWithParticipant`
- `countAnnotations`

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get all the orthography tokens in UC427_ViktoriaPapp_A_ENG.eaf
orthography <- getAnnotations(labbcat.url, "UC427_ViktoriaPapp_A_ENG.eaf", "orthography")

## Get the first 20 orthography tokens in UC427_ViktoriaPapp_A_ENG.eaf
orthography <- getAnnotations(labbcat.url, "UC427_ViktoriaPapp_A_ENG.eaf", "orthography", 20, 0)

## End(Not run)
```
getAvailableMedia

List the media available for the given transcript.

Description
List the media available for the given transcript.

Usage
getAvailableMedia(labbcat.url, id)

Arguments
labbcat.url URL to the LaBB-CAT instance
id A transcript ID (i.e. transcript name)

Value
A named list of media files available for the given transcript, with members:

- `trackSuffix` The track suffix of the media
- `mimeType` The MIME type of the file
- `url` URL to the content of the file
- `name` Name of the file

See Also
getTranscriptIds

Examples
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## List the media files available for BR2044_OllyOhlson.eaf
media <- getAvailableMedia(labbcat.url, "BR2044_OllyOhlson.eaf")

## End(Not run)
getCorpusIds

*Gets a list of corpus IDs.*

**Description**

Returns a list of corpora in the given 'LaBB-CAT' instance.

**Usage**

```r
getCorpusIds(labcat.url)
```

**Arguments**

- `labcat.url` URL to the LaBB-CAT instance

**Value**

A list of corpus IDs

**Examples**

```r
## Not run:
## List corpora
corpora <- getCorpusIds("https://labcat.canterbury.ac.nz/demo/"

## End(Not run)
```

---

getDeserializerDescriptors

*Lists the descriptors of all registered deserializers.*

**Description**

Returns a list of deserializers, which are modules that import transcriptions and annotation structures from a specific file format, e.g. Praat TextGrid, plain text, etc.

**Usage**

```r
getDeserializerDescriptors(labcat.url)
```

**Arguments**

- `labcat.url` URL to the LaBB-CAT instance
Value

A list of serializers, each including the following information:

- **name** The name of the format.
- **version** The installed version of the serializer module.
- **fileSuffixes** The normal file name suffixes (extensions) of the files.
- **mimeType** The MIME type of the format, i.e. the value to use as the `mimeType` parameter of `getFragments`.

Examples

```r
## Not run:
## List file upload formats supported
formats <- getDeserializerDescriptors("https://labbcat.canterbury.ac.nz/demo/")

## can we upload as plain text?
plainTextSupported <- "text/plain" %in% formats$mimeType

## End(Not run)
```
getDictionaryEntries  

Usage

getDictionaryEntries(labbcat.url, manager.id, dictionary.id, keys)

Arguments

  labbcat.url  URL to the LaBB-CAT instance
  manager.id   The layer manager ID of the dictionary, as returned by getDictionaries
  dictionary.id  The ID of the dictionary, as returned by getDictionaries
  keys         A list of entries to look up

Value

A data frame with the keys and their dictionary entries.

See Also

getDictionaries

dictionaries <- getDictionaries("https://labbcat.canterbury.ac.nz/demo/")

dictionaries <- getDictionaries("https://labbcat.canterbury.ac.nz/demo/")

getDictionaryEntries(labbcat.url, manager.id, dictionary.id, keys)

dictionaries <- getDictionaries("https://labbcat.canterbury.ac.nz/demo/")

dictionaries <- getDictionaries("https://labbcat.canterbury.ac.nz/demo/")

## Not run:
## List the dictionaries available
dictionaries <- getDictionaries("https://labbcat.canterbury.ac.nz/demo/")

## End(Not run)

getDictionaryEntries  

Lookup entries in a dictionary.

Examples

## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/

keys <- c("the", "quick", "brown", "fox")

## get the pronunciations according to CELEX
entries <- getDictionaryEntries(labbcat.url, "CELEX-EN", "Phonology (wordform)", keys)

## End(Not run)
getFragments

**getFragments**

Gets fragments transcript in a given format.

**Description**

This function gets fragments of transcripts from 'LaBB-CAT', converted to a given format (by default, Praat TextGrid).

**Usage**

```r
getFragments(
  labbcat.url,
  id,
  start,
  end,
  layer.ids,
  mime.type = "text/praat-textgrid",
  path = ""
)
```

**Arguments**

- **labbcat.url**: URL to the LaBB-CAT instance
- **id**: The transcript ID (transcript name) of the sound recording, or a vector of transcript IDs.
- **start**: The start time in seconds, or a vector of start times.
- **end**: The end time in seconds, or a vector of end times.
- **layer.ids**: A vector of layer IDs.
- **mime.type**: Optional content-type - "text/praat-textgrid" is the default, but your LaBB-CAT installation may support other formats, which can be discovered using `getSerializerDescriptors`.
- **path**: Optional path to directory where the files should be saved.

**Details**

*NB* Although many formats will generate exactly one file for each interval (e.g. `mime.type=text/praat-textgrid`), this is not guaranteed; some formats generate a single file or a fixed collection of files regardless of how many fragments there are.

**Value**

The name of the file, which is saved in the current directory, or a list of names of files, if multiple id/s/start/s/end/s were specified.

If a list of files is returned, they are in the order that they were returned by the server, which *should* be the order that they were specified in the id/start/end lists.
getGraphIds

Deprecated synonym for getTranscriptIds.

Description

Returns a list of graph IDs (i.e. transcript names).

Usage

getGraphIds(labbcat.url)

Arguments

labbcat.url URL to the LaBB-CAT instance

Value

A list of graph IDs

See Also

getTranscriptIds
getGraphIdsInCorpus

Examples

```r
## Not run:
## List all transcripts
transcripts <- getGraphIds("https://labbcat.canterbury.ac.nz/demo/")

## End(Not run)
```

getGraphIdsInCorpus  Deprecated synonym for getTranscriptIdsInCorpus.

Description

Returns a list of corpora in the given 'LaBB-CAT' instance.

Usage

```r
getGraphIdsInCorpus(labbcat.url, id)
```

Arguments

- `labbcat.url` URL to the LaBB-CAT instance
- `id` The ID (name) of the corpus

Value

A list of corpus IDs

See Also

`getGraphIdsInCorpus`

Examples

```r
## Not run:
## List transcripts in the QB corpus
transcripts <- getGraphIdsInCorpus("https://labbcat.canterbury.ac.nz/demo/", "QB")

## End(Not run)
```
getGraphIdsWithParticipant

*Deprecated synonym for getTranscriptIdsWithParticipant.*

**Description**

Returns a list of IDs of graphs (i.e. transcript names) that include the given participant.

**Usage**

`getGraphIdsWithParticipant(labbcat.url, id)`

**Arguments**

- `labbcat.url`  
  URL to the LaBB-CAT instance
- `id`  
  A participant ID

**Value**

A list of graph IDs

**See Also**

`getTranscriptIdsWithParticipant`

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## List transcripts in which UC427_ViktoriaPapp_A_ENG speaks
transcripts <- getGraphIdsWithParticipant(labbcat.url, "UC427_ViktoriaPapp_A_ENG")

## End(Not run)
```

getId

*Gets the store’s ID.*

**Description**

The store’s ID - i.e. the ID of the 'LaBB-CAT' instance.

**Usage**

`getId(labbcat.url)`
getLayer

Arguments

labbcat.url URL to the LaBB-CAT instance

Value

The annotation store’s ID

Examples

## Not run:
## Get ID of LaBB-CAT instance
instance.id <- getId("https://labbcat.canterbury.ac.nz/demo/")

## End(Not run)

getLayer gets a layer definition.

Description

Gets a layer definition.

Usage

getLayer(labbcat.url, id)

Arguments

labbcat.url URL to the LaBB-CAT instance
id ID of the layer to get the definition for

Value

The definition of the given layer, with members:

- id The layer’s unique ID
- parentId The layer’s parent layer ID
- description The description of the layer
- alignment The layer’s alignment - 0 for none, 1 for point alignment, 2 for interval alignment
- peers Whether children have peers or not
- peersOverlap Whether child peers can overlap or not
- parentIncludes Whether the parent t-includes the child
- saturated Whether children must temporally fill the entire parent duration (true) or not (false)
- parentIncludes Whether the parent t-includes the child
- type The type for labels on this layer
- validLabels List of valid label values for this layer
getLayerIds

See Also
getLayerIds, getLayers

Examples

## Not run:
## Get the definition of the orthography layer
orthography.layer <- getLayer("https://labbcat.canterbury.ac.nz/demo/", "orthography")

## End(Not run)

getLayerIds

Gets a list of layer IDs.

Description

Layer IDs are annotation 'types'.

Usage

getLayerIds(labbcat.url)

Arguments

labbcat.url URL to the LaBB-CAT instance

Value

A list of layer IDs

Examples

## Not run:
## Get names of all layers
layer.ids <- getLayerIds("https://labbcat.canterbury.ac.nz/demo/"

## End(Not run)
getLayers

Description

Gets a list of layer definitions.

Usage

getLayers(labbcat.url)

Arguments

labbcat.url URL to the LaBB-CAT instance

Value

A list of layer definitions, with members:

- id The layer’s unique ID
- parentId The layer’s parent layer ID
- description The description of the layer
- alignment The layer’s alignment - 0 for none, 1 for point alignment, 2 for interval alignment
- peers Whether children have peers or not
- peersOverlap Whether child peers can overlap or not
- parentIncludes Whether the parent t-includes the child
- saturated Whether children must temporally fill the entire parent duration (true) or not (false)
- parentIncludes Whether the parent t-includes the child
- type The type for labels on this layer
- validLabels List of valid label values for this layer

See Also

getLayerIds

Examples

## Not run:
## Get definitions of all layers
layers <- getLayers("https://labbcat.canterbury.ac.nz/demo/")

## End(Not run)
getMatchAlignments

Gets temporal alignments of matches on a given layer.

Description

Gets labels and start/end offsets of annotations on a given layer, identified by given match IDs.

Usage

```r
getMatchAlignments(
  labbcat.url,  # URL to the LaBB-CAT instance
  match.ids,    # A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.
  layer.ids,    # A vector of layer IDs.
  target.offset = 0,  # The distance from the original target of the match, e.g.
    • 0 – find annotations of the match target itself,
    • 1 – find annotations of the token immediately after match target
    • -1 – find annotations of the token immediately before match target
  annotations.per.layer = 1,  # The number of annotations on the given layer to retrieve. In most cases, there’s only one annotation available. However, tokens may, for example, be annotated with ‘all possible phonemic transcriptions’, in which case using a value of greater than 1 for this parameter provides other phonemic transcriptions, for tokens that have more than one.
  anchor.confidence.min = 50,  # The minimum confidence for alignments, e.g.
    • 0 – return all alignments, regardless of confidence;
    • 50 – return only alignments that have been at least automatically aligned;
    • 100 – return only manually-set alignments.
  include.match.ids = FALSE,  # Whether or not the data frame returned includes the original MatchId column or not.
  page.length = 1000,
  no.progress = FALSE)
```

Arguments

- `labbcat.url` URL to the LaBB-CAT instance
- `match.ids` A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.
- `layer.ids` A vector of layer IDs.
- `target.offset` The distance from the original target of the match, e.g.
  - 0 - find annotations of the match target itself,
  - 1 - find annotations of the token immediately after match target
  - -1 - find annotations of the token immediately before match target
- `annotations.per.layer` The number of annotations on the given layer to retrieve. In most cases, there’s only one annotation available. However, tokens may, for example, be annotated with ‘all possible phonemic transcriptions’, in which case using a value of greater than 1 for this parameter provides other phonemic transcriptions, for tokens that have more than one.
- `anchor.confidence.min` The minimum confidence for alignments, e.g.
  - 0 - return all alignments, regardless of confidence;
  - 50 - return only alignments that have been at least automatically aligned;
  - 100 - return only manually-set alignments.
- `include.match.ids` Whether or not the data frame returned includes the original MatchId column or not.
getMatches

In order to prevent timeouts when there are a large number of matches or the network connection is slow, rather than retrieving matches in one big request, they are retrieved using many smaller requests. This parameter controls the number of results retrieved per request.

FALSE to suppress visual progress bar. Otherwise, progress bar will be shown when interactive().

Details

You can specify a threshold for confidence in the alignment, which is a value from 0 (not aligned) to 100 (manually aligned). The default is 50 (automatically aligned), so only alignments that have been at least automatically aligned are specified. For cases where there's a token but its alignment confidence falls below the threshold, a label is returned, but the start/end times are NA.

Value

A data frame with label, start time, and end time, for each layer.

See Also

getMatches getMatchLabels

Examples

## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Perform a search
results <- getMatches(labbcat.url, list(segment="I"))

## Get the segment following the token, with alignment if it's been manually aligned
following.segment <- getMatchAlignments(labbcat.url, results$MatchId, "segment",
  target.offset=1, anchor.confidence.min=100)

## End(Not run)
getMatches

Usage

getMatches(
  labbcat.url,
  pattern,
  participant.ids = NULL,
  transcript.types = NULL,
  main.participant = TRUE,
  aligned = FALSE,
  matches.per.transcript = NULL,
  words.context = 0,
  max.matches = NULL,
  overlap.threshold = NULL,
  page.length = 1000,
  no.progress = FALSE
)

Arguments

labbcat.url URL to the LaBB-CAT instance

pattern An object representing the pattern to search for.

Strictly speaking, this should be a named list that replicates the structure of the ‘search matrix’ in the LaBB-CAT browser interface, with one element called “columns”, containing a named list for each column.

Each element in the “columns” named list contains an element named “layers”, whose value is a named list for patterns to match on each layer, and optionally an element named “adj”, whose value is a number representing the maximum distance, in tokens, between this column and the next column - if “adj” is not specified, the value defaults to 1, so tokens are contiguous.

Each element in the “layers” named list is named after the layer it matches, and the value is a named list with the following possible elements:

- `pattern` A regular expression to match against the label
- `min` An inclusive minimum numeric value for the label
- `max` An exclusive maximum numeric value for the label
- `not` TRUE to negate the match
- `anchorStart` TRUE to anchor to the start of the annotation on this layer (i.e. the matching word token will be the first at/after the start of the matching annotation on this layer)
- `anchorEnd` TRUE to anchor to the end of the annotation on this layer (i.e. the matching word token will be the last before/at the end of the matching annotation on this layer)
- `target` TRUE to make this layer the target of the search; the results will contain one row for each match on the target layer

Examples of valid pattern objects include:

```r
## words starting with 'ps...'
pattern <- list(columns = list(
  ps = list(pattern = '/\b\w+\b/s',
    min = 1, max = 5)
))
```
list(layers = list(
  orthography = list(pattern = "ps.*"))))

## the word 'the' followed immediately or with one intervening word by
## a hapax legomenon (word with a frequency of 1) that doesn't start with a vowel
pattern <- list(columns = list(
  list(layers = list(
    orthography = list(pattern = "the")),
    adj = 2),
  list(layers = list(
    phonemes = list(not = TRUE, pattern = "[cCEFHiIPqujV0123456789~#\$@].*"),
    frequency = list(max = "2"))))

For ease of use, the function will also accept the following abbreviated forms:

## a single list representing a 'one column' search,
## and string values, representing regular expression pattern matching
pattern <- list(orthography = "ps.*")

## a list containing the columns (adj defaults to 1, so matching tokens are contiguous).
pattern <- list(
  list(orthography = "the"),
  list(phonemes = list(not = TRUE, pattern = "[cCEFHiIPqujV0123456789~#\$@].*"),
  frequency = list(max = "2"))))

participant.ids
An optional list of participant IDs to search the utterances of. If not supplied, all utterances in the corpus will be searched.

transcript.types
An optional list of transcript types to limit the results to. If null, all transcript types will be searched.

main.participant
TRUE to search only main-participant utterances, FALSE to search all utterances.

aligned
true to include only words that are aligned (i.e. have anchor confidence &ge; 50, false to search include un-aligned words as well.

matches.per.transcript
Optional maximum number of matches per transcript to return. NULL means all matches.

words.context
Number of words context to include in the ‘Before.Match’ and ‘After.Match’ columns in the results.

max.matches
The maximum number of matches to return, or null to return all.

overlap.threshold
The percentage overlap with other utterances before simultaneous speech is excluded, or null to include overlapping speech.

page.length
In order to prevent timeouts when there are a large number of matches or the network connection is slow, rather than retrieving matches in one big request, they are retrieved using many smaller requests. This parameter controls the number of results retrieved per request.
getMatches

no.progress TRUE to suppress visual progress bar. Otherwise, progress bar will be shown when interactive().

Value

A data frame identifying matches, containing the following columns:

• SearchName A name based on the pattern – the same for all rows
• MatchId A unique ID for the matching target token
• Transcript Name of the transcript in which the match was found
• Participant Name of the speaker
• Corpus The corpus of the transcript
• Line The start offset of the utterance/line
• LineEnd The end offset of the utterance/line
• Before.Match Transcript text immediately before the match
• Text Transcript text of the match
• After.Match Transcript text immediately after the match
• Number Row number
• URL URL of the first matching word token
• Target.word Text of the target word token
• Target.word.start Start offset of the target word token
• Target.word.end End offset of the target word token
• Target.segment Label of the target segment (only present if the segment layer is included in the pattern)
• Target.segment.start Start offset of the target segment (only present if the segment layer is included in the pattern)
• Target.segment.end End offset of the target segment (only present if the segment layer is included in the pattern)

See Also

getParticipantIds

Examples

## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## create a pattern object to match against
pattern <- list(columns = list(
  list(layers = list(
    orthography = list(pattern = "the"),
    adj = 2),
  list(layers = list(
    orthography = list(pattern = "the")},
```
getMatchingGraphIds

Deprecated synonym for getMatchingTranscriptIds.

Description

Gets a list of IDs of graphs (i.e. transcript names) that match a particular pattern.

Usage

def getMatchingGraphIds(
    labbcat.url,
    expression,
    page.length = NULL,
    page.number = NULL,
    order = NULL
)

Arguments

labbcat.url URL to the LaBB-CAT instance
testpattern An expression that determines which graphs match
testpattern The maximum number of IDs to return, or null to return all
testpattern The zero-based page number to return, or null to return the first page
testpattern An expression that determines the order the graphs are listed in - if specified, this must include the keyword 'ASC' for ascending or 'DESC' for descending order.

details

The results can be exhaustive, by omitting pageLength and page.number, or they can be a subset (a 'page') of results, by given pageLength and page.number values.

The order of the list can be specified. If omitted, the graphs are listed in ID order.

The expression language is currently not well defined, but is based on JavaScript syntax.
• The `labels` function can be used to represent a list of all the annotation labels on a given layer. For example, each transcript can have multiple participants, so the participant labels (names) are represented by: `labels('participant')`

• Use the `includes` function on a list to test whether the list contains a given element. e.g. to match transcripts that include the participant 'Joe' use: `labels('participant').includes('Joe')`

• Use the `first` function to identify the first (or the only) annotation on a given layer. e.g. the annotation representing the transcript’s corpus is: `first('corpus')`

• Single annotations have various attributes, including 'id', 'label', 'ordinal', etc. e.g. the name of the transcript’s corpus is: `first('corpus').label`

• Regular expressions can be matched by using expressions like `'/regex/.test(str)'`, e.g. to test if the ID starts with 'BR' use: `/^BR.+/.test(id)` or to test if the transcript’s corpus includes a B use: `/.*B.*/.test(first('corpus').label)`

Expressions such as those in the examples can be used.

Value

A list of graph IDs (i.e. transcript names)

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get all transcripts whose names start with "BR"
transcripts <- getMatchingGraphIds(labbcat.url, "/^BR.+/.test(id)"

## Get the first twenty transcripts in the "QB" corpus
transcripts <- getMatchingGraphIds(labbcat.url, "first('corpus').label = 'QB'", 20, 0)

## Get the second transcript that has "QB247_Jacqui" as a speaker
transcripts <- getMatchingGraphIds(labbcat.url, "labels('participant').includes('QB247_Jacqui')", 1, 1)

## Get all transcripts in the QB corpus whose names start with "BR"
## in word-count order
transcripts <- getMatchingGraphIds(labbcat.url, "first('corpus').label = 'QB' & & /^BR.+/.test(id)", order="first('transcript_word_count').label ASC")

## End(Not run)
```
getMatchingParticipantIds

Gets a list of IDs of participants that match a particular pattern.

Description

Gets a list of IDs of participants that match a particular pattern.

Usage

getMatchingParticipantIds(
    labcat.url,
    expression,
    page.length = NULL,
    page.number = NULL
)

Arguments

labcat.url URL to the LaBB-CAT instance
expression An expression that determines which participants match
page.length The maximum number of IDs to return, or null to return all
page.number The zero-based page number to return, or null to return the first page

Details

The results can be exhaustive, by omitting page.length and page.number, or they can be a subset (a 'page') of results, by given page.length and page.number values.

The expression language is currently not well defined, but is based on JavaScript syntax.

- The labels function can be used to represent a list of all the annotation labels on a given layer. For example, each participant can have multiple corpora, so the corpus labels (names) are represented by: labels('corpus')
- Use the includes function on a list to test whether the list contains a given element. e.g. to match participants that include the corpus 'QB' use: labels('corpus').includes('QB')
- Use the first function to identify the first (or the only) annotation on a given layer. e.g. the annotation representing the participant’s gender is: first('participant_gender')
- Single annotations have various attributes, including 'id', 'label', 'ordinal', etc. e.g. the label of the participant’s gender is: first('participant_gender').label
- Regular expressions can be matched by using expressions like '/regex/.test(str)', e.g. to test if the ID starts with 'BR' use: /^BR.+/.test(id) or to test if the participant’s gender includes 'binary' use: /.+binary.*/.test(first('participant_gender').label)

Expressions such as those in the examples can be used.
getMatchingTranscriptIds

**Value**

A list of participant IDs

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get all participants whose IDs start with "BR"
participants <- getMatchingParticipantIds(labbcat.url, "/^BR.+/.test(id)"

## Get the first twenty transcripts in the "QB" corpus
participants <- getMatchingParticipantIds(
  labbcat.url, "labels('corpus').includes('QB')", 20, 0)

## Get all participants in the "QB" corpus that have "Jacqui" as part of the ID
participants <- getMatchingTranscriptParticipantIds(
  labbcat.url, "labels('corpus').includes('QB') && /^BR.+/.test(id)"

## End(Not run)
```

---

**getMatchingTranscriptIds**

*Gets a list of IDs of transcripts that match a particular pattern.*

**Description**

Gets a list of IDs of transcripts (i.e. transcript names) that match a particular pattern.

**Usage**

```r
getMatchingTranscriptIds(
  labbcat.url, expression, page.length = NULL, page.number = NULL, order = NULL)
```

**Arguments**

- `labbcat.url` URL to the LaBB-CAT instance
- `expression` An expression that determines which transcripts match
- `page.length` The maximum number of IDs to return, or null to return all
- `page.number` The zero-based page number to return, or null to return the first page
order

An expression that determines the order the transcripts are listed in. If specified, this must include the keyword 'ASC' for ascending or 'DESC' for descending order.

Details

The results can be exhaustive, by omitting page.length and page.number, or they can be a subset (a 'page') of results, by given page.length and page.number values.

The order of the list can be specified. If omitted, the transcripts are listed in ID order.

The expression language is currently not well defined, but is based on JavaScript syntax.

- The labels function can be used to represent a list of all the annotation labels on a given layer. For example, each transcript can have multiple participants, so the participant labels (names) are represented by: labels('participant')

- Use the includes function on a list to test whether the list contains a given element. e.g. to match transcripts that include the participant 'Joe' use: labels('participant').includes('Joe')

- Use the first function to identify the first (or the only) annotation on a given layer. e.g. the annotation representing the transcript’s corpus is: first('corpus')

- Single annotations have various attributes, including 'id', 'label', 'ordinal', etc. e.g. the name of the transcript’s corpus is: first('corpus').label

- Regular expressions can be matched by using expressions like '/regex/.test(str)', e.g. to test if the ID starts with 'BR' use: /^BR.+/ .test(id) or to test if the transcript’s corpus includes a B use: /.*B.*/.test(first('corpus').label)

Expressions such as those in the examples can be used.

Value

A list of transcript IDs (i.e. transcript names)

Examples

```
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get all transcripts whose names start with "BR"
transcripts <- getMatchingTranscriptIds(labbcat.url, "/^BR.+/\.test(id)"

## Get the first twenty transcripts in the "QB" corpus
transcripts <- getMatchingTranscriptIds(
    labbcat.url, "first('corpus').label == 'QB', 20, 0"

## Get the second transcript that has "QB247_Jacqui" as a speaker
transcripts <- getMatchingTranscriptIds(
    labbcat.url, "labels('participant').includes('QB247_Jacqui')", 1, 1"

## Get all transcripts in the QB corpus whose names start with "BR"
## in word-count order
```
getMatchLabels <- getMatchingTranscriptIds(
  labbcat.url, "first('corpus').label = 'QB' && "BR.+/test(id)",
  order="first('transcript_word_count').label ASC")

## End(Not run)

getMatchLabels

Gets labels of annotations on a given layer, identified by given match IDs.

Description

Gets labels of annotations on a given layer, identified by given match IDs.

Usage

getMatchLabels(
  labbcat.url, 
  match.ids, 
  layer.ids, 
  target.offset = 0, 
  annotations.per.layer = 1, 
  include.match.ids = FALSE, 
  page.length = 1000, 
  no.progress = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>labbcat.url</td>
<td>URL to the LaBB-CAT instance</td>
</tr>
<tr>
<td>match.ids</td>
<td>A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.</td>
</tr>
<tr>
<td>layer.ids</td>
<td>A vector of layer IDs.</td>
</tr>
<tr>
<td>target.offset</td>
<td>The distance from the original target of the match, e.g.</td>
</tr>
<tr>
<td>annotations.per.layer</td>
<td>The number of annotations on the given layer to retrieve. In most cases, there’s only one annotation available. However, tokens may, for example, be annotated with ‘all possible phonemic transcriptions’, in which case using a value of greater than 1 for this parameter provides other phonemic transcriptions, for tokens that have more than one.</td>
</tr>
<tr>
<td>include.match.ids</td>
<td>Whether or not the data frame returned includes the original MatchId column or not.</td>
</tr>
</tbody>
</table>
getMedia

In order to prevent timeouts when there are a large number of matches or the network connection is slow, rather than retrieving matches in one big request, they are retrieved using many smaller requests. This parameter controls the number of results retrieved per request.

No progress

TRUE to suppress visual progress bar. Otherwise, progress bar will be shown when interactive().

Value

A data frame of labels.

See Also

getMatches getMatchAlignments

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Perform a search
results <- getMatches(labbcat.url, list(orthography="quake"))

## Get the topic annotations for the matches
topics <- getMatchLabels(labbcat.url, results$MatchId, "topic")

## End(Not run)
```

getMedia

Gets a given media track for a given transcript.

Description

Gets a given media track for a given transcript.

Usage

getMedia(labbcat.url, id, track.suffix = "", mime.type = "audio/wav")

Arguments

- `labbcat.url`: URL to the LaBB-CAT instance
- `id`: A transcript ID (i.e. transcript name)
- `track.suffix`: The track suffix of the media
- `mime.type`: The MIME type of the media
getMediaTracks

List the predefined media tracks available for transcripts.

Description

List the predefined media tracks available for transcripts.

Usage

gメディアTracks(labbcat.url)

Arguments

labbcat.url URL to the LaBB-CAT instance

Value

A list of media track definitions.

Examples

## Not run:
## Get the media tracks configured in LaBB-CAT
tracks <- getMediaTracks("https://labcat.canterbury.ac.nz/demo/")

## End(Not run)
getParticipantAttributes

*Gets participant attribute values for given participant IDs.*

### Description

Gets participant attribute values for given participant IDs.

### Usage

```r
getParticipantAttributes(labbcat.url, participant.ids, layer.ids)
```

### Arguments

- **labbcat.url**: URL to the LaBB-CAT instance
- **participant.ids**: A vector of participant IDs
- **layer.ids**: A vector of layer IDs corresponding to participant attributes. In general, these are layers whose ID is prefixed `participant_`, however formally it's any layer where `layer$parentId == 'participant'` & `layer$alignment == 0`.

### Value

A data frame of attribute value labels.

### Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get gender and age for all participants
attributes <- getParticipantAttributes(labbcat.url,
  getParticipantIds(labbcat.url),
  c('participant_gender', 'participant_age'))
```

## End(Not run)
getParticipantIds

*Gets a list of participant IDs.*

**Description**

Returns a list of participant IDs.

**Usage**

```r
getParticipantIds(labbcat.url)
```

**Arguments**

- `labbcat.url` URL to the LaBB-CAT instance

**Value**

A list of participant IDs

**Examples**

```r
## Not run:
## List all speakers
speakers <- getParticipantIds("https://labbcat.canterbury.ac.nz/demo/")
## End(Not run)
```

getSerializerDescriptors

*Lists the descriptors of all registered serializers.*

**Description**

Returns a list of serializers, which are modules that export annotation structures as a specific file format, e.g. Praat TextGrid, plain text, etc., so the `mimeType` of descriptors reflects what `mimeTypes` can be specified for `getFragments`.

**Usage**

```r
getSerializerDescriptors(labbcat.url)
```

**Arguments**

- `labbcat.url` URL to the LaBB-CAT instance
getSoundFragments

Value

A list of serializers, each including the following information:

- **name** The name of the format.
- **version** The installed version of the serializer module.
- **fileSuffixes** The normal file name suffixes (extensions) of the files,
- **mimeType** The MIME type of the format, i.e. the value to use as the *mimeType* parameter of `getFragments`.

See Also

getFragments

Examples

```r
## Not run:
## List file export formats supported
formats <- getSerializerDescriptors("https://labbcat.canterbury.ac.nz/demo/")

## can we export as plain text?
plainTextSupported <- "text/plain" %in% formats$mimeType

## End(Not run)
```

---

**getSoundFragments**

*Gets sound fragments from 'LaBB-CAT'.*

Description

Gets sound fragments from 'LaBB-CAT'.

Usage

```r
getSoundFragments(
  labbcat.url,
  ids,
  start.offsets,
  end.offsets,
  sample.rate = NULL,
  path = "",
  no.progress = FALSE
)
```
getSoundFragments

Arguments

labbcat.url URL to the LaBB-CAT instance
ids The transcript ID (transcript name) of the sound recording, or a vector of transcript IDs.
start.offsets The start time in seconds, or a vector of start times.
end.offsets The end time in seconds, or a vector of end times.
sample.rate Optional sample rate in Hz - if a positive integer, then the result is a mono file with the given sample rate.
pure Optional path to directory where the files should be saved.
no.progress TRUE to supress visual progress bar. Otherwise, progress bar will be shown when interactive().

Value

The name of the file, which is saved in the current directory, or a list of names of files, if multiple id's/start's/end’s were specified

If a list of files is returned, they are in the order that they were returned by the server, which *should* be the order that they were specified in the id/start/end lists.

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get the 5 seconds starting from 10s after the beginning of a recording
wav.file <- getSoundFragments(labbcat.url, "AP2505_Nelson.eaf", 10.0, 15.0, path="samples")

## Get the 5 seconds starting from 10s as a mono 22kHz file
wav.file <- getSoundFragments(labbcat.url, "AP2505_Nelson.eaf", 10.0, 15.0, 22050)

## Load some search results previously exported from LaBB-CAT
results <- read.csv("results.csv", header=T)

## Get a list of fragments
wav.files <- getSoundFragments(labbcat.url, results$Transcript, results$Line, results$LineEnd)

## Get a list of fragments
wav.file <- getSoundFragments(labbcat.url, results$Transcript, results$Line, results$LineEnd)

## End(Not run)
```
### getSystemAttribute

*Gets the value of the given system attribute.*

#### Description

Gets the value of the given system attribute.

#### Usage

```r
getSystemAttribute(labbcat.url, attribute)
```

#### Arguments

- `labbcat.url` URL to the LaBB-CAT instance
- `attribute` Name of the attribute.

#### Value

The value of the given attribute.

`getLayers`

#### Examples

```r
## Not run:
## Get the name of the LaBB-CAT instance
title <- getSystemAttribute("https://labbcat.canterbury.ac.nz/demo/", "title")
## End(Not run)
```

### getTranscriptAttributes

*Gets transcript attribute values for given transcript IDs.*

#### Description

Gets transcript attribute values for given transcript IDs.

#### Usage

```r
getTranscriptAttributes(labbcat.url, transcript.ids, layer.ids)
```
getTranscriptIds

Arguments

labbcat.url  URL to the LaBB-CAT instance
transcript.ids A vector of transcript IDs
layer.ids A vector of layer IDs corresponding to transcript attributes. In general, these are layers whose ID is prefixed 'transcript_', however formally it’s any layer where layer$parentId == 'transcript' && layer$alignment == 0, which includes 'corpus' as well as transcript attribute layers.

Value

A data frame of attribute value labels.

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get language, duration, and corpus for transcripts starting with 'BR'
attributes <- getTranscriptAttributes(labbcat.url,
    getMatchingTranscriptIds(labbcat.url, "'/BR.+/.test(id)"),
    c('transcript_language', 'transcript_duration', 'corpus'))
```

## End(Not run)

getTranscriptIds  Gets a list of transcript IDs.

Description

Returns a list of transcript IDs (i.e. transcript names).

Usage

getTranscriptIds(labbcat.url)

Arguments

labbcat.url  URL to the LaBB-CAT instance

Value

A list of transcript IDs
Examples

```r
## Not run:
## List all transcripts
transcripts <- getTranscriptIds("https://labbcat.canterbury.ac.nz/demo/")

## End(Not run)
```

---

getTranscriptIdsInCorpus

*Gets a list of transcript in a corpus.*

Description

Returns a list of transcript IDs in the given corpus.

Usage

```r
getTranscriptIdsInCorpus(labbcat.url, id)
```

Arguments

- `labbcat.url`: URL to the LaBB-CAT instance
- `id`: The ID (name) of the corpus

Value

A list of transcript IDs

Examples

```r
## Not run:
## List transcripts in the QB corpus
transcripts <- getTranscriptIdsInCorpus("https://labbcat.canterbury.ac.nz/demo/", "QB")

## End(Not run)
```
getTranscriptIdsWithParticipant

*Gets a list of IDs of transcripts that include the given participant.*

**Description**

Returns a list of IDs of transcripts (i.e. transcript names) that include the given participant.

**Usage**

```
getTranscriptIdsWithParticipant(labbcat.url, id)
```

**Arguments**

- `labbcat.url`: URL to the LaBB-CAT instance
- `id`: A participant ID

**Value**

A list of transcript IDs

**See Also**

`getParticipantIds`

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## List transcripts in which UC427_ViktoriaPapp_A_ENG speaks
transcripts <- getTranscriptIdsWithParticipant(labbcat.url, "UC427_ViktoriaPapp_A_ENG")

## End(Not run)
```

getUserInfo

*Gets information about the current user.*

**Description**

Returns information about the current user, including the roles or groups they are in.

**Usage**

```
getUserInfo(labbcat.url)
```
labbcatCredentials

Arguments

  labbcat.url  URL to the LaBB-CAT instance

Value

A named list containing information about current the LaBB-CAT user.

See Also

labbcatCredentials

Examples

## Not run:
## List file export formats supported
me <- getUserInfo("https://labbcat.canterbury.ac.nz/demo/")

## am I an administrator?
admin <- "admin" %in% me$roles

## End(Not run)

labbcatCredentials  Sets the username and password that the package should use for connecting to a given LaBB-CAT server in future function calls.

Description

This step is optional, as all functions will prompt the user for the username and password if required. If the script is running in RStudio, then the RStudio password input dialog is used, hiding the credentials from view. Otherwise, the console is used, and credentials are visible.

Usage

labbcatCredentials(labbcat.url, username, password)

Arguments

  labbcat.url  URL to the LaBB-CAT instance
  username  The LaBB-CAT username, if it is password-protected
  password  The LaBB-CAT password, if it is password-protected

Details

The recommended approach is to *not* use labbcatCredentials, to avoid saving user credentials in script files that may eventually become visible to other. Use labbcatCredentials *only* in cases where the script execution is unsupervised.
Value

NULL if the username/password are correct, and a string describing the problem if a problem occurred, e.g. "Credentials rejected" if the username/password are incorrect, or a string starting "Version mismatch" if the server's version of LaBB-CAT is lower than the minimum required.

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## specify the username/password in the script
## (only use labbcatCredentials for scripts that must execute unsupervised!)
labbcatCredentials(labbcat.url, "demo", "demo")

## End(Not run)
```

---

**labbcatTimeout**

Sets the timeout for request to the LaBB-CAT server in future function calls. The default timeout is 10 seconds.

Description

Sets the timeout for request to the LaBB-CAT server in future function calls. The default timeout is 10 seconds.

Usage

```r
labbcatTimeout(seconds = NULL)
```

Arguments

- `seconds`  
The number of seconds before requests return with a timeout error.

Value

The request timeout in seconds

Examples

```r
## Not run:
## the request timeout
labbcatTimeout(30)

## End(Not run)
```
newTranscript

**Description**

This function adds a transcript and optionally a media file to the corpus.

**Usage**

```r
newTranscript(
  labbcat.url,
  transcript,
  media = NULL,
  transcript.type = NULL,
  corpus = NULL,
  episode = NULL,
  no.progress = FALSE
)
```

**Arguments**

- `labbcat.url` URL to the LaBB-CAT instance
- `transcript` The path to the transcript to upload.
- `media` The path to the media to upload, if any.
- `transcript.type` The transcript type.
- `corpus` The corpus to add the transcript to.
- `episode` The transcript’s episode.
- `no.progress` TRUE to supress visual progress bar. Otherwise, progress bar will be shown when interactive().

**Details**

For this function to work, the credentials used to connect to the server must have at least 'edit' access.

**Value**

The ID of the new transcript in the corpus

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"
```
## Get attributes for new transcript

corpus <- getCorpusIds(labbcat.url)[1]
transcript.type.layer <- getLayer(labbcat.url, "transcript_type")
transcript.type <- transcript.type.layer$validLabels[[1]]

## upload transcript

newTranscript(
  labbcat.url, "my-transcript.eaf", "my-transcript.wav",
  "", transcript.type, corpus, "episode-1")

## End(Not run)

---

**nzilbb.labbcat**  
*Accessing Data Stored in 'LaBB-CAT' Instances*

### Description

'LaBB-CAT' is a web-based language corpus management system developed by the New Zealand Institute of Language, Brain and Behaviour (NZILBB) - see <https://labbcat.canterbury.ac.nz>. This package defines functions for accessing corpus data in a 'LaBB-CAT' instance. You must have at least version 20210601.1528 of 'LaBB-CAT' to use this package. For more information about 'LaBB-CAT', see Robert Fromont and Jennifer Hay (2008) <doi:10.3366/E1749503208000142> or Robert Fromont (2017) <doi:10.1016/j.csl.2017.01.004>.

### Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>nzilbb.labbcat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version:</td>
<td>1.0-1</td>
</tr>
<tr>
<td>Date:</td>
<td>2021-07-07</td>
</tr>
<tr>
<td>Title:</td>
<td>Accessing Data Stored in 'LaBB-CAT' Instances</td>
</tr>
<tr>
<td>Authors@R:</td>
<td>c(person(&quot;Robert&quot;, &quot;Fromont&quot;, role = c(&quot;aut&quot;, &quot;cre&quot;), email = &quot;<a href="mailto:robert.fromont@canterbury.ac.nz">robert.fromont@canterbury.ac.nz</a>&quot;, comment = &quot;&quot;))</td>
</tr>
<tr>
<td>Imports:</td>
<td>jsonlite, httr, stringr, utils, rstudioapi</td>
</tr>
<tr>
<td>Description:</td>
<td>'LaBB-CAT' is a web-based language corpus management system developed by the New Zealand Institute of Language, Brain and Behaviour, University of Canterbury</td>
</tr>
<tr>
<td>License:</td>
<td>GPL (&gt;= 3)</td>
</tr>
<tr>
<td>Copyright:</td>
<td>New Zealand Institute of Language, Brain and Behaviour, University of Canterbury</td>
</tr>
<tr>
<td>URL:</td>
<td><a href="https://nzilbb.github.io/labbcat-R/">https://nzilbb.github.io/labbcat-R/</a>, <a href="https://labbcat.canterbury.ac.nz">https://labbcat.canterbury.ac.nz</a></td>
</tr>
<tr>
<td>RoxygenNote:</td>
<td>7.1.1</td>
</tr>
<tr>
<td>Suggests:</td>
<td>testthat (&gt;= 2.1.0)</td>
</tr>
<tr>
<td>Author:</td>
<td>Robert Fromont [aut, cre] (<a href="https://orcid.org/0000-0001-5271-5487">https://orcid.org/0000-0001-5271-5487</a>)</td>
</tr>
<tr>
<td>Maintainer:</td>
<td>Robert Fromont <a href="mailto:robert.fromont@canterbury.ac.nz">robert.fromont@canterbury.ac.nz</a></td>
</tr>
</tbody>
</table>

### Index of help topics:

- **countAnnotations**  
  Gets the number of annotations on the given layer of the given transcript.
- **deleteTranscript**  
  Delete a transcript from the corpus.
getAllUtterances Get all utterances of participants.
getAnchors Gets the given anchors in the given transcript.
getAnnotations Gets the annotations on the given layer of the given transcript.
getAvailableMedia List the media available for the given transcript.
getCorpusIds Gets a list of corpus IDs.
getDeserializerDescriptors Lists the descriptors of all registered deserializers.
getDictionaries List the dictionaries available.
getDictionaryEntries Lookup entries in a dictionary.
getFragments Gets fragments transcript in a given format.
getGraphIds Deprecated synonym for getTranscriptIds.
getGraphIdsInCorpus Deprecated synonym for getTranscriptIdsInCorpus.
getGraphIdsWithParticipant Deprecated synonym for getTranscriptIdsWithParticipant.
getId Gets the store's ID.
getLayer Gets a layer definition.
getLayerIds Gets a list of layer IDs.
getLayers Gets a list of layer definitions.
getMatchAlignments Gets temporal alignments of matches on a given layer.
getMatchLabels Gets labels of annotations on a given layer, identified by given match IDs.
getMatches Search for tokens.
getMatchGraphIds Deprecated synonym for getMatchingTranscriptIds.
getMatchingParticipantIds Gets a list of IDs of participants that match a particular pattern.
getMatchingTranscriptIds Gets a list of IDs of transcripts that match a particular pattern.
getMedia Gets a given media track for a given transcript.
getMediaTracks List the predefined media tracks available for transcripts.
getParticipantAttributes Gets participant attribute values for given participant IDs.
getParticipantIds Gets a list of participant IDs.
getSerializerDescriptors Lists the descriptors of all registered serializers.
getSoundFragments Gets sound fragments from 'LaBB-CAT'.
getSystemAttribute

Gets the value of the given system attribute.

getTranscriptAttributes

Gets transcript attribute values for given transcript IDs.

getTranscriptIds

Gets a list of transcript IDs.

getTranscriptIdsInCorpus

Gets a list of transcript in a corpus.

getTranscriptIdsWithParticipant

Gets a list of IDs of transcripts that include the given participant.

getUserInfo

Gets information about the current user.

labbbcatCredentials

Sets the username and password that the package should use for connecting to a given LaBB-CAT server in future function calls.

labbbcatTimeout

Sets the timeout for request to the LaBB-CAT server in future function calls. The default timeout is 10 seconds.

newTranscript

Upload a new transcript.

nzilbb.labbbcat

Accessing Data Stored in 'LaBB-CAT' Instances

praatScriptCentreOfGravity

Generates a script for extracting the CoG, for use with processWithPraat.

praatScriptFastTrack

Generates a script for extracting formants using FastTrack, for use with processWithPraat.

praatScriptFormants

Generates a script for extracting formants, for use with processWithPraat.

praatScriptIntensity

Generates a script for extracting maximum intensity, for use with processWithPraat.

praatScriptPitch

Generates a script for extracting pitch, for use with processWithPraat.

processWithPraat

Process a set of intervals with Praat.

updateTranscript

Update an existing transcript.

'LaBB-CAT' is a web-based language corpus management system and this package provides access to data stored in a 'LaBB-CAT' instance. You must have at least version 20210601.1528 'LaBB-CAT' to use this package.

Author(s)

NA

References


Examples

## Not run:
praatScriptCentreOfGravity

Generates a script for extracting the CoG, for use with processWithPraat.

Description

This function generates a Praat script fragment which can be passed as the praat.script parameter of processWithPraat, in order to extract one or more spectral centre of gravity (CoG) measurements.

Usage

praatScriptCentreOfGravity(powers = c(2), spectrum.fast = TRUE)

Arguments

powers A vector of numbers specifying which powers to query for to extract, e.g. c(1.0,2.0).
spectrum.fast Whether to use the ‘fast’ option when creating the spectrum object to query.

Value

A script fragment which can be passed as the praat.script parameter of processWithPraat

See Also

processWithPraat
praatScriptFormants
praatScriptIntensity
praatScriptPitch
praatScriptFastTrack
praatScriptFastTrack

Generates a script for extracting formants using FastTrack, for use with processWithPraat.

Example usage:
```r
praatScriptFastTrack(
  formants = c(1, 2),
  sample.points = c(0.5),
  lowest.analysis.frequency = 5000,
  lowest.analysis.frequency.male = 4500,
  highest.analysis.frequency = 7000,
  highest.analysis.frequency.male = 6500,
  gender.attribute = "participant_gender",
  value.for.male = "M",
  time.step = 0.002,
  tracking.method = "burg",
  number.of.formants = 3,
  maximum.f1.frequency = 1200,
  maximum.f1.bandwidth = NULL,
  maximum.f2.bandwidth = NULL,
  maximum.f3.bandwidth = NULL,
  minimum.f4.frequency = 2900,
  enable.rhotic.heuristic = TRUE,
  enable.f3.f4.proximity.heuristic = TRUE,
  number.of.steps = 20,
  number.of.coefficients = 5
)
```

Description

This function generates a Praat script fragment which can be passed as the praat.script parameter of processWithPraat, in order to extract selected formants using the FastTrack Praat plugin.
Arguments

formants  A vector of integers specifying which formants to extract, e.g. c(1,2) for the first and second formant.

sample.points  A vector of numbers (0 <= sample.points <= 1) specifying multiple points at which to take the measurement. The default is a single point at 0.5 - this means one measurement will be taken halfway through the target interval. If, for example, you wanted eleven measurements evenly spaced throughout the interval, you would specify sample.points as being c(0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0).

lowest.analysis.frequency
  Lowest analysis frequency (Hz) by default.

lowest.analysis.frequency.male
  Lowest analysis frequency (Hz) for male speakers, or NULL to use the same value as lowest.analysis.frequency.

highest.analysis.frequency
  Highest analysis frequency (Hz) by default.

highest.analysis.frequency.male
  Highest analysis frequency (Hz) for male speakers, or NULL to use the same value as highest.analysis.frequency.

gender.attribute
  Name of the LaBB-CAT participant attribute that contains the participant’s gender - normally this is "participant_gender".

value.for.male
  The value that the gender.attribute has when the participant is male.

time.step
  Time step in seconds.

tracking.method
  tracking_method parameter for trackAutoselectProcedure; "burg" or "robust".

number.of.formants
  Number of formants to track - 3 or 4.

maximum.f1.frequency
  Specifying a non-NULL value enables the F1 frequency heuristic: Median F1 frequency should not be higher than this value.

maximum.f1.bandwidth
  Specifying a non-NULL value (e.g. 500) enables the F1 bandwidth heuristic: Median F1 bandwidth should not be higher than this value.

maximum.f2.bandwidth
  Specifying a non-NULL value (e.g. 600) enables the F2 bandwidth heuristic: Median F2 bandwidth should not be higher than this value.

maximum.f3.bandwidth
  Specifying a non-NULL value (e.g. 900) enables the F3 bandwidth heuristic: Median F3 bandwidth should not be higher than this value.

minimum.f4.frequency
  Specifying a non-NULL value enables the F4 frequency heuristic: Median F4 frequency should not be lower than this value.
enable.rhotic.heuristic
   Whether to enable the rhotic heuristic: If F3 < 2000 Hz, F1 and F2 should be at least 500 Hz apart.

enable.f3.f4.proximity.heuristic
   Whether to enable the F3/F4 proximity heuristic: If (F4 - F3) < 500 Hz, F1 and F2 should be at least 1500 Hz apart.

number.of.steps
   Number of analyses between low and high analysis limits. More analysis steps may improve results, but will increase analysis time (50 percent more steps = around 50 percent longer to analyze).

number.of.coefficients
   Number of coefficients for formant prediction. More coefficients allow for more sudden, and 'wiggly' formant motion.

Details

The FastTrack Praat plugin, developed by Santiago Barreda, automatically runs multiple formant analyses on each segment, selects the best (the smoothest, with optional heuristics), and makes the winning formant object available for measurement. For more information, see https://github.com/santiagobarreda/FastTrack

Value

A script fragment which can be passed as the praat.script parameter of processWithPraat

See Also

   processWithPraat
   praatScriptCentreOfGravity
   praatScriptIntensity
   praatScriptPitch
   praatScriptFormants

Examples

   ## Not run:
   ## define the LaBB-CAT URL
   labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

   ## Get all tokens of the KIT vowel
   results <- getMatches(labbcat.url, list(segment="I"))

   ## Get the first 3 formants at three points during the vowel
   formants <- processWithPraat(
      labbcat.url,
      results$MatchId, results$Target.segment.start, results$Target.segment.end,
      window.offset=0.025,
      praatScriptFastTrack(formants=c(1,2,3),
      sample.points=c(0.25,0.5,0.75)))
praatScriptFormants

Generates a script for extracting formants, for use with processWithPraat.

**Description**

This function generates a Praat script fragment which can be passed as the praat.script parameter of processWithPraat, in order to extract selected formants.

**Usage**

```r
praatScriptFormants(
  formants = c(1, 2),
  sample.points = c(0.5),
  time.step = 0,
  max.number.formants = 5,
  max.formant = 5500,
  max.formant.male = 5000,
  gender.attribute = "participant_gender",
  value.for.male = "M",
  window.length = 0.025,
  preemphasis.from = 50
)
```

**Arguments**

- **formants** A vector of integers specifying which formants to extract, e.g c(1, 2) for the first and second formant.
- **sample.points** A vector of numbers (0 <= sample.points <= 1) specifying multiple points at which to take the measurement. The default is a single point at 0.5 - this means one measurement will be taken halfway through the target interval. If, for example, you wanted eleven measurements evenly spaced throughout the interval, you would specify sample.points as being c(0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0).
- **time.step** Time step in seconds, or 0.0 for 'auto'.
- **max.number.formants** Maximum number of formants.
- **max.formant** Maximum formant value (Hz) for all speakers, or for female speakers, if max.formant.male is also specified.
- **max.formant.male** Maximum formant value (Hz) for male speakers, or NULL to use the same value as max.formant.
**praatScriptFormants**

- **gender.attribute**  
  Name of the LaBB-CAT participant attribute that contains the participant’s gender - normally this is "participant_gender".

- **value.for.male**  
  The value that the gender.attribute has when the participant is male.

- **window.length**  
  Window length in seconds.

- **preemphasis.from**  
  Pre-emphasis from (Hz)

**Details**

The `praatScriptFastTrack` function provides an alternative to this function which uses the FastTrack Praat plugin for formant analysis.

**Value**

A script fragment which can be passed as the praat.script parameter of `processWithPraat`

**See Also**

- `processWithPraat`
- `praatScriptCentreOfGravity`
- `praatScriptIntensity`
- `praatScriptPitch`
- `praatScriptFastTrack`

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Get all tokens of the KIT vowel
results <- getMatches(labbcat.url, list(segment="I"))

## Get the first 3 formants at three points during the vowel
formants <- processWithPraat(
  labbcat.url,
  results$MatchId, results$Target.segment.start, results$Target.segment.end,
  window.offset=0.025,
  praatScriptFormants(formants=c(1,2,3),
                      sample.points=c(0.25,0.5,0.75)))

## End(Not run)
```
praatScriptIntensity

Generates a script for extracting maximum intensity, for use with processWithPraat.

Description

This function generates a Praat script fragment which can be passed as the praat.script parameter of processWithPraat, in order to extract maximum intensity value.

Usage

praatScriptIntensity(minimum.pitch = 100, time.step = 0, subtract.mean = TRUE)

Arguments

- minimum.pitch: Minimum pitch (Hz).
- time.step: Time step in seconds, or 0.0 for 'auto'.
- subtract.mean: Whether to subtract the mean or not.

Value

A script fragment which can be passed as the praat.script parameter of processWithPraat

See Also

- processWithPraat
- praatScriptFormants
- praatScriptCentreOfGravity
- praatScriptPitch
- praatScriptFastTrack

Examples

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Perform a search
results <- getMatches(labbcat.url, list(segment="s"))

## Get intensity for all matches
intensity <- processWithPraat(
  labbcat.url,
  results$MatchId, results$Target.segment.start, results$Target.segment.end,
  praatScriptIntensity())

## End(Not run)
```
praatScriptPitch

Generates a script for extracting pitch, for use with processWithPraat.

Description

This function generates a Praat script fragment which can be passed as the praat.script parameter of processWithPraat, in order to extract pitch information.

Usage

praatScriptPitch(
  get.mean = TRUE,
  get.minimum = FALSE,
  get.maximum = FALSE,
  time.step = 0,
  pitch.floor = 60,
  max.number.of.candidates = 15,
  very.accurate = FALSE,
  silence.threshold = 0.03,
  voicing.threshold = 0.5,
  octave.cost = 0.01,
  octave.jump.cost = 0.35,
  voiced.unvoiced.cost = 0.35,
  pitch.ceiling = 500,
  pitch.floor.male = 30,
  voicing.threshold.male = 0.4,
  pitch.ceiling.male = 250,
  gender.attribute = "participant_gender",
  value.for.male = "M",
  window.length = 0.025,
  preemphasis.from = 50
)

Arguments

- **get.mean**: Extract the mean pitch for the sample.
- **get.minimum**: Extract the minimum pitch for the sample.
- **get.maximum**: Extract the maximum pitch for the sample.
- **time.step**: Step setting for praat command
- **pitch.floor**: Minimum pitch (Hz) for all speakers, or for female speakers, if pitch.floor.male is also specified.
- **max.number.of.candidates**: Maximum number of candidates setting for praat command
- **very.accurate**: Accuracy setting for praat command
silence.threshold
   Silence threshold setting for praat command
voicing.threshold
   Voicing threshold (Hz) for all speakers, or for female speakers, if voicing.threshold.male is also specified.
octave.cost
   Octave cost setting for praat command
doctave.jump.cost
   Octave jump cost setting for praat command
voiced.unvoiced.cost
   Voiced/unvoiced cost setting for praat command
pitch.ceiling
   Maximum pitch (Hz) for all speakers, or for female speakers, if pitch.floor.male is also specified.
pitch.floor.male
   Minimum pitch (Hz) for male speakers.
voicing.threshold.male
   Voicing threshold (Hz) for male speakers.
pitch.ceiling.male
   Maximum pitch (Hz) for male speakers.
gender.attribute
   Name of the LaBB-CAT participant attribute that contains the participant’s gender - normally this is "participant_gender".
value.for.male
   The value that the gender.attribute has when the participant is male.
window.length
   Window length in seconds.
preemphasis.from
   Pre-emphasis from (Hz)

Value

   A script fragment which can be passed as the praat.script parameter of processWithPraat

See Also

   processWithPraat
   praatScriptFormants
   praatScriptCentreOfGravity
   praatScriptIntensity
   praatScriptFastTrack

Examples

   ## Not run:
   ## define the LaBB-CAT URL
   labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

   ## Perform a search
   results <- getMatches(labbcat.url, list(segment="I"))
## Get pitch mean, max, and min for all matches
pitch <- processWithPraat(
  labbcat.url,
  results$MatchId, results$Target.segment.start, results$Target.segment.end,
  praatScriptPitch(get.mean=TRUE, get.minimum=TRUE, get.maximum=TRUE))

## End(Not run)

**processWithPraat**  
*Process a set of intervals with Praat.*

### Description
This function instructs the LaBB-CAT server to invoke Praat for a set of sound intervals, in order to extract acoustic measures.

### Usage
```
processWithPraat(
  labbcat.url,  
  match.ids,   
  start.offsets,  
  end.offsets,   
  praat.script,  
  window.offset = 0,  
  gender.attribute = "participant_gender",  
  attributes = NULL,  
  no.progress = FALSE
)
```

### Arguments
- **labbcat.url**: URL to the LaBB-CAT instance
- **match.ids**: A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.
- **start.offsets**: The start time in seconds, or a vector of start times.
- **end.offsets**: The end time in seconds, or a vector of end times.
- **praat.script**: Script to run on each match. This may be a single string or a character vector.
- **window.offset**: In many circumstances, you will want some context before and after the sample start/end time. For this reason, you can specify a "window offset" - this is a number of seconds to subtract from the sample start and add to the sample end time, before extracting that part of the audio for processing. For example, if the sample starts at 2.0s and ends at 3.0s, and you set the window offset to 0.5s, then Praat will extract a sample of audio from 1.5s to 3.5s, and do the selected processing on that sample.
gender.attribute

Which participant attribute represents the participant’s gender.

attributes

Vector of participant attributes to make available to the script. For example, if you want to use different acoustic parameters depending on what the gender of the speaker is, including the "participant_gender" attribute will make a variable called participant_gender$ available to the praat script, whose value will be the gender of the speaker for that segment.

no.progress

TRUE to supress visual progress bar. Otherwise, progress bar will be shown when interactive().

Details

The exact measurements to return depend on the praat.script that is invoked. This is a Praat script fragment that will run once for each sound interval specified.

There are functions to allow the generation of a number of pre-defined praat scripts for common tasks such as formant, pitch, intensity, and centre of gravity – see praatScriptFormants, praatScriptCentreOfGravity, praatScriptIntensity and praatScriptPitch.

You can provide your own script, either by building a string with your code, or loading one from a file.

LaBB-CAT prefixes praat.script with code to open a sound file and extract a defined part of it into a Sound object which is then selected.

LaBB-CAT ‘Remove’s this Sound object after the script finishes executing. Any other objects created by the script must be ‘Remove’d before the end of the script (otherwise Praat runs out of memory during very large batches)

LaBB-CAT assumes that all calls to the function 'print' correspond to fields for export and each field must be printed on its own line. Specifically it scans for lines of the form:

print 'myOutputVariable' 'newline$

Variables that can be assumed to be already set in the context of the script are:

- windowOffset – the value used for the Window Offset; how much context to include.
- windowAbsoluteStart – the start time of the window extracted relative to the start of the original audio file.
- windowAbsoluteEnd – the end time of the window extracted relative to the start of the original audio file.
- windowDuration – the duration of the window extracted (including window offset).
- targetAbsoluteStart – the start time of the target interval relative to the start of the original audio file.
- targetAbsoluteEnd – the end time of the target interval relative to the start of the original audio file.
- targetStart – the start time of the target interval relative to the start of the window extracted.
- targetEnd – the end time of the target interval relative to the start of the window extracted.
- targetDuration – the duration of the target interval.
- sampleNumber – the number of the sample within the set of samples being processed.
- sampleName$ – the name of the extracted/selected Sound object.
processWithPraat

Value

A data frame of acoustic measures, one row for each matchId.

See Also

praatScriptFormants
praatScriptCentreOfGravity
praatScriptIntensity
praatScriptPitch

Examples

## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/"

## Perform a search
results <- getMatches(labbcat.url, list(segment="I"))

## get F1 and F2 for the mid point of the vowel
formants <- processWithPraat(
  labbcat.url,
  results$MatchId, results$Target.segment.start, results$Target.segment.end,
  praatScriptFormants())

## get first 3 formants at three points during the sample, the mean, min, and max
## pitch, the max intensity, and the CoG using powers 1 and 2
acoustic.measurements <- processWithPraat(
  labbcat.url,
  results$MatchId, results$Target.segment.start, results$Target.segment.end,
  paste(
    praatScriptFormants(c(1,2,3), c(0.25,0.5,0.75)),
    praatScriptPitch(get.mean=TRUE, get.minimum=TRUE, get.maximum=TRUE),
    praatScriptIntensity(),
    praatScriptCentreOfGravity(powers=c(1.0,2.0)),
    window.offset=0.5)

## execute a custom script loaded form a file
acoustic.measurements <- processWithPraat(
  labbcat.url,
  results$MatchId, results$Target.segment.start, results$Target.segment.end,
  readLines("acousticMeasurements.praat"))

## End(Not run)
**updateTranscript**

Update an existing transcript.

**Description**

This function uploads a new version of an existing transcript.

**Usage**

\[
\text{updateTranscript}(\text{labbcat.url, transcript.path, no.progress = FALSE})
\]

**Arguments**

- `labbcat.url`: URL to the LaBB-CAT instance
- `transcript.path`: The path to the transcript to upload.
- `no.progress`: TRUE to suppress visual progress bar. Otherwise, progress bar will be shown when interactive().

**Details**

For this function to work, the credentials used to connect to the server must have at least 'edit' access.

**Value**

The ID of the updated transcript in the corpus

**Examples**

```r
## Not run:
## define the LaBB-CAT URL
labbcat.url <- "https://labbcat.canterbury.ac.nz/demo/

## upload new version of transcript transcript
updateTranscript(labbcat.url, "my-transcript.eaf")
```

## End(Not run)
Index

* anchor
  getAnchors, 6

* annotation
  getMatchAlignments, 20
  getMatchLabels, 30
  getParticipantAttributes, 33
  getTranscriptAttributes, 37

* audio
  getAvailableMedia, 9
  getMedia, 31

* connect
  getUserInfo, 40
  labbcatCredentials, 41
  labbcatTimeout, 42

* corpora
  getCorpusIds, 10
  getGraphIdsInCorpus, 15
  getTranscriptIdsInCorpus, 39

* corpus
  getGraphIdsInCorpus, 15
  getTranscriptIdsInCorpus, 39

* dictionary
  getDictionaries, 11
  getDictionaryEntries, 12

* expression
  getMatchingGraphIds, 25
  getMatchingParticipantIds, 27
  getMatchingTranscriptIds, 28

* format
  getDeserializerDescriptors, 10
  getSerializerDescriptors, 34

* fragment
  getFragments, 13
  getSoundFragments, 35

* graph
  getGraphIdsWithParticipant, 16
  getMatchingGraphIds, 25
  getTranscriptIdsWithParticipant, 40

* label
  getMatchAlignments, 20
  getMatchLabels, 30
  getParticipantAttributes, 33
  getTranscriptAttributes, 37

* layer
  getLayer, 17
  getLayerIds, 18
  getLayers, 19
  getMatchAlignments, 20
  getMatchLabels, 30
  getParticipantAttributes, 33
  getTranscriptAttributes, 37

* management
  deleteTranscript, 4
  newTranscript, 43
  updateTranscript, 59

* media
  getAvailableMedia, 9
  getMedia, 31
  getMediaTracks, 32

* package
  nzilbb.labbcat, 44

* participant
  getParticipantIds, 34

* password
  labbcatCredentials, 41
  labbcatTimeout, 42

* praat
  praatScriptCentreOfGravity, 47
  praatScriptFastTrack, 48
  praatScriptFormants, 51
  praatScriptIntensity, 53
  praatScriptPitch, 54
  processWithPraat, 56

* sample
  getFragments, 13
  getSoundFragments, 35

* search
getAllUtterances, 4
getMatches, 21

* sound
getFragments, 13
getMediaTracks, 32
getSoundFragments, 35

* speaker
getParticipantIds, 34

* timeout
labbcatCredentials, 41
labbcatTimeout, 42

* transcript
countAnnotations, 3
deleteTranscript, 4
getAnnotations, 7
getGraphIds, 14
getGraphIdsWithParticipant, 16
getMatchingGraphIds, 25
getMatchingParticipantIds, 27
getMatchingTranscriptIds, 28
getTranscriptIds, 38
getTranscriptIdsWithParticipant, 40
newTranscript, 43
updateTranscript, 59

* username
getUserInfo, 40
labbcatCredentials, 41
labbcatTimeout, 42

* wav
getFragments, 13
getSoundFragments, 35

countAnnotations, 3, 8
deleteTranscript, 4

getAllUtterances, 4
getAnchors, 6
getAnnotations, 7, 7
getAvailableMedia, 9
getCorpusIds, 10
getDeserializerDescriptors, 10
getDictionaries, 11, 12
getDictionaryEntries, 11, 12
getFragments, 11, 13, 34, 35
getGraphIds, 14
getGraphIdsInCorpus, 15, 15
getGraphIdsWithParticipant, 16
getId, 16
getLayer, 17
getLayerIds, 18, 18, 19
getLayers, 18, 19, 37
getMatchAlignments, 20, 31
getMatches, 21, 21, 31
getMatchingGraphIds, 25
getMatchingParticipantIds, 27
getMatchingTranscriptIds, 28
getMatchLabels, 21, 30
getMedia, 31
getMediaTracks, 32
getParticipantAttributes, 33
getParticipantIds, 6, 24, 34, 40
getSerializerDescriptors, 13, 14, 34
getSoundFragments, 35
getSystemAttribute, 37
get TranscriptAttributes, 37
getTranscriptIds, 3, 8, 9, 14, 32, 38
getTranscriptIdsInCorpus, 3, 8, 39
getTranscriptIdsWithParticipant, 3, 8, 16, 40
getUserInfo, 40

labbcatCredentials, 41, 41
labbcatTimeout, 42

newTranscript, 43
nziob.labbcat, 44

praatScriptCentreOfGravity, 47, 50, 52, 53, 55, 57, 58
praatScriptFastTrack, 47, 48, 52, 53, 55
praatScriptFormants, 47, 50, 51, 53, 55, 57, 58
praatScriptIntensity, 47, 50, 52, 53, 55, 57, 58
praatScriptPitch, 47, 50, 52, 53, 54, 57, 58
processWithPraat, 47, 48, 50–55, 56

updateTranscript, 59