Package 'nzilbb.labbcat'

August 27, 2020

Version 0.6-1

Date 2020-08-25

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 20200812.1253 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>.

License GPL (>= 3)

Copyright New Zealand Institute of Language, Brain and Behaviour, University of Canterbury

URL https://github.com/nzilbb/labbcat-R, https://labbcat.canterbury.ac.nz

RoxygenNote 7.1.1

Suggests testthat (>= 2.1.0)

NeedsCompilation no

Author Robert Fromont [aut, cre]

Maintainer Robert Fromont <robert@fromont.net.nz>

Repository CRAN

Date/Publication 2020-08-27 00:50:02 UTC

R topics documented:

`countAnnotations` *Gets the number of annotations on the given layer of the given transcript.*

Description

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

Usage

```
countAnnotations(labbcat.url, id, layerId)
```

Arguments

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

Value

The number of annotations on that layer

See Also

`getTranscriptIds` `getTranscriptIdsInCorpus` `getTranscriptIdsWithParticipant`

Examples

```
## 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)
```

getAnchors *Gets the given anchors in the given transcript.*

Description

Lists the given anchors in the given transcript.

Usage

```
getAnchors(labbcat.url, id, anchorId)
```

Arguments

labbcat.url	URL to the LaBB-CAT instance
id	A transcript ID (i.e. transcript name)
anchorId	A vector of anchor IDs (or a string representing one anchor ID)

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

```
## 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, layerId, pageLength = NULL, pageNumber = NULL)
```

Arguments

`labbcat.url`	URL to the LaBB-CAT instance
`id`	A transcript ID (i.e. transcript name)
`layerId`	A layer ID
`pageLength`	The maximum number of annotations to return, or null to return all
`pageNumber`	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

```
## 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

```
getCorpusIds(labbcat.url)
```

Arguments

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

Value

A list of corpus IDs

Examples

```
## Not run:
## List corpora
corpora <- getCorpusIds("https://labbcat.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

```
getDeserializerDescriptors(labbcat.url)
```

Arguments

`labbcat.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

```
## 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)
```

getDictionaries *List the dictionaries available.*

Description

List the dictionaries available.

Usage

```
getDictionaries(labbcat.url)
```

Arguments

labbcat.url	URL to the LaBB-CAT instance

Value

A named list of layer manager IDs, each of which containing a list of dictionaries that the layer manager makes available.

See Also

getDictionaryEntries

Examples

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

## End(Not run)
```

getDictionaryEntries *Lookup entries in a dictionary.*

Description

Lookup entries in a dictionary.

Usage

```
getDictionaryEntries(labbcat.url, managerId, dictionaryId, keys)
```

Arguments

labbcat.url	URL to the LaBB-CAT instance
managerId	The layer manager ID of the dictionary, as returned by getDictionaries
dictionaryId	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

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 *Gets fragments of annotation transcript from 'LaBB-CAT', converted to a given format (by default, Praat TextGrid)*

Description

Gets fragments of annotation transcript from 'LaBB-CAT', converted to a given format (by default, Praat TextGrid)

Usage

```
getFragments(
  labbcat.url,
  id,
  start,
  end,
  layerIds,
  mimeType = "text/praat-textgrid",
  no.progress = FALSE,
  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.
layerIds	A vector of layer IDs.
mimeType	Optional content-type - "text/praat-textgrid" is the default, but your LaBB-CAT installation may support other formats, which can be discovered using getSerializerDescriptors.
no.progress	Optionally suppress the progress bar when multiple fragments are specified - TRUE for no progress bar.
path	Optional path to directory where the files should be saved.

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.

See Also

getSerializerDescriptors

Examples

```
## 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
textgrid.file <- getFragments(labbcat.url, "AP2505_Nelson.eaf", 10.0, 15.0,
    c("transcript", "phonemes"), path="samples")

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

## Get a list of fragment TextGrids, including the utterances, transcript, and phonemes layers
textgrid.files <- getFragments(
    labbcat.url, results$Transcript, results$Line, results$LineEnd,
    c("utterances", "transcript", "phonemes"))

## Get a list of fragment TextGrids with no progress bar
textgrid.files <- getFragments(
    labbcat.url, results$Transcript, results$Line, results$LineEnd, no.progress=TRUE)

## End(Not run)
```

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

Examples

```
## 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

```
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

```
## 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

```
## 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)
```

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

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 *Gets a list of layer definitions.*

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

```
getMatchAlignments(
  labbcat.url,
  matchIds,
  layerIds,
  targetOffset = 0,
  annotationsPerLayer = 1,
  anchor.confidence.min = 50
)
```

Arguments

`labbcat.url`	URL to the LaBB-CAT instance
`matchIds`	A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.
`layerIds`	A vector of layer IDs.
`targetOffset`	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
`annotationsPerLayer`	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.

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(segments="I"))

## Get the segment following the token, with alignment if it's been manually aligned
following.segment <- getMatchAlignments(labbcat.url, results$MatchId, "segments",
```

```
    targetOffset=1, anchor.confidence.min=100)

## End(Not run)
```

`getMatches` *Search for tokens.*

Description

Searches through transcripts for tokens matching the given pattern.

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,
  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:

```
## words starting with 'ps...'
pattern <- list(columns = list(
    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 = "[cCEFHiIPqQuUV0123456789~#$@].*"),
           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 = "[cCEFHiIPqQuUV0123456789~#$@].*"),
         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 ≥ 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.

`no.progress`
: Optionally suppress the progress bar when multiple fragments are specified - TRUE for no progress bar.

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.transcript* Text of the target word token
- *Target.transcript.start* Start offset of the target word token
- *Target.transcript.end* End offset of the target word token
- *Target.segments* Label of the target segment (only present if the segment layer is included in the pattern)
- *Target.segments.start* Start offset of the target segment (only present if the segment layer is included in the pattern)
- *Target.segments.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(
            phonemes = list(not=TRUE, pattern = "[cCEFHiIPqQuUV0123456789~#\$@].*"),
            frequency = list(max = "2")))))

## get the tokens matching the pattern
results <- getMatches(labbcat.url, pattern)

## results$MatchId can be used to access results

## End(Not run)
```

`getMatchingGraphIds` *Deprecated synonym for getMatchingTranscriptIds.*

Description

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

Usage

```
getMatchingGraphIds(
  labbcat.url,
  expression,
  pageLength = NULL,
  pageNumber = NULL,
  order = NULL
)
```

Arguments

`labbcat.url`	URL to the LaBB-CAT instance
`expression`	An expression that determines which graphs match
`pageLength`	The maximum number of IDs to return, or null to return all
`pageNumber`	The zero-based page number to return, or null to return the first page
`order`	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 pageNumber, or they can be a subset (a 'page') of results, by given pageLength and pageNumber values.

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

The expression language is currently not well defined, but expressions such as those in the examples can be used.

Value

A list of graph 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 <- getMatchingGraphIds(labbcat.url, "id MATCHES 'BR.+'")

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

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

## Get all transcripts whose names start with "BR" and have "QB247_Jacqui" as a speaker,
## in word-count order
transcripts <- getMatchingGraphIds(
     labbcat.url, "my('corpus').label = 'QB' AND 'QB247_Jacqui' IN labels('participant')", 1, 1,
        "my('transcript_word_count').label ASC")

## 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

```
getMatchingTranscriptIds(
  labbcat.url,
  expression,
  pageLength = NULL,
  pageNumber = NULL,
  order = NULL
)
```

Arguments

labbcat.url	URL to the LaBB-CAT instance
expression	An expression that determines which transcripts match
pageLength	The maximum number of IDs to return, or null to return all
pageNumber	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 pageLength and pageNumber, or they can be a subset (a 'page') of results, by given pageLength and pageNumber values.

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

The expression language is currently not well defined, but 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, "id MATCHES 'BR.+'")

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

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

## Get all transcripts whose names start with "BR" and have "QB247_Jacqui" as a speaker,
## in word-count order
transcripts <- getMatchingTranscriptIds(
     labbcat.url, "my('corpus').label = 'QB' AND 'QB247_Jacqui' IN labels('participant')", 1, 1,
        "my('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,
  matchIds,
  layerIds,
  targetOffset = 0,
  annotationsPerLayer = 1
)
```

Arguments

labbcat.url	URL to the LaBB-CAT instance
matchIds	A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.
layerIds	A vector of layer IDs.
targetOffset	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
annotationsPerLayer	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.

Value

A data frame of labels.

See Also

getMatches getMatchAlignments

Examples

```
## 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, trackSuffix = "", mimeType = "audio/wav")
```

Arguments

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

Value

A URL to the given media for the given transcript

See Also

getTranscriptIds

Examples

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

## Get URL for the WAV file for BR2044_OllyOhlson.eaf
media <- getMedia(labbcat.url, "BR2044_OllyOhlson.eaf")
```

```
## Get URL for the 'QuakeFace' video file for BR2044_OllyOhlson.eaf
media <- getMedia(labbcat.url, "BR2044_OllyOhlson.eaf", "_face", "video/mp4")

## End(Not run)
```

`getMediaTracks` *List the predefined media tracks available for transcripts.*

Description

List the predefined media tracks available for transcripts.

Usage

```
getMediaTracks(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://labbcat.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

```
getParticipantAttributes(labbcat.url, participantIds, layerIds)
```

Arguments

labbcat.url	URL to the LaBB-CAT instance
participantIds	A vector of participant IDs
layerIds	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

```
## 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

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

Arguments

labbcat.url	URL to the LaBB-CAT instance

Value

A list of participant IDs

Examples

```
## 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 *mimeType*s can be specified for getFragments.

Usage

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

Arguments

`labbcat.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,

See Also

getFragments

Examples

```
## 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

```
getSoundFragments(
  labbcat.url,
  ids,
  startOffsets,
  endOffsets,
  sampleRate = NULL,
  no.progress = FALSE,
  path = ""
)
```

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.
`startOffsets`	The start time in seconds, or a vector of start times.
`endOffsets`	The end time in seconds, or a vector of end times.
`sampleRate`	Optional sample rate in Hz - if a positive integer, then the result is a mono file with the given sample rate.
`no.progress`	Optionally suppress the progress bar when multiple fragments are specified - TRUE for no progress bar.
`path`	Optional path to directory where the files should be saved.

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

```
## 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 with no progress bar
wav.file <- getSoundFragments(
         labbcat.url, results$Transcript, results$Line, results$LineEnd, no.progress=TRUE)

## End(Not run)
```

---

getSystemAttribute *Gets the value of the given system attribute.*

Description

Gets the value of the given system attribute.

Usage

```
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

```
## 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

```
getTranscriptAttributes(labbcat.url, transcriptIds, layerIds)
```

Arguments

`labbcat.url`	URL to the LaBB-CAT instance
`transcriptIds`	A vector of transcript IDs
`layerIds`	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$parentId == 'graph' && layer$alignment == 0, which includes 'corpus' as well as transcript attribute layers.

Value

A data frame of attribute value labels.

Examples

```
## 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, "id MATCHES 'BR.+'"),
```

```
            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

```
## 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

```
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

```
## 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

```
## 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)
```

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

```
## 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

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

Arguments

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

Value

The request timeout in seconds

Examples

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

## 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 20200812.1253 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

Package:	nzilbb.labbcat
Version:	0.6-1
Date:	2020-08-25
Title:	Accessing Data Stored in 'LaBB-CAT' Instances
Authors@R:	c(person("Robert", "Fromont", role = c("aut", "cre"), email = "robert@fromont.net.nz"))
Imports:	jsonlite, httr, stringr, utils, rstudioapi
Description:	'LaBB-CAT' is a web-based language corpus management system developed by the New Zealand Institute o
License:	GPL (>= 3)
Copyright:	New Zealand Institute of Language, Brain and Behaviour, University of Canterbury
URL:	https://github.com/nzilbb/labbcat-R, https://labbcat.canterbury.ac.nz
RoxygenNote:	7.1.1
Suggests:	testthat (>= 2.1.0)
Author:	Robert Fromont [aut, cre]
Maintainer:	Robert Fromont <robert@fromont.net.nz>

Index of help topics:

```
getMatchLabels          Gets labels of annotations on a given layer,
                        identified by given match IDs.
getMatches              Search for tokens.
getMatchingGraphIds     Deprecated synonym for
                        getMatchingTranscriptIds.
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.
labbcatCredentials      Sets the username and password that the package
                        should use for connecting to a given LaBB-CAT
                        server in future function calls.
labbcatTimeout          Sets the timeout for request to the LaBB-CAT
                        server in future function calls. The default
                        timeout is 10 seconds.
nzilbb.labbcat          Accessing Data Stored in 'LaBB-CAT' Instances
praatScriptCentreOfGravity
                        Generates a script for extracting the CoG, 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.
```

'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 20200608.1507 of 'LaBB-CAT' to use this package.

Author(s)

NA

References

Robert Fromont and Jennifer Hay, "ONZE Miner: the development of a browser-based research tool", 2008 Robert Fromont, "Toward a format-neutral annotation store", 2017

Examples

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

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

## Get the phonemic transcriptions for the matches
phonemes <- getMatchLabels(labbcat.url, results$MatchId, "phonemes")

## Get sound fragments for the matches
wav.files <- getSoundFragments(labbcat.url, results$Transcript, results$Line, results$LineEnd)

## End(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

Examples

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

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

## Get centres of gravity for all matches
cog <- processWithPraat(
              labbcat.url,
            results$MatchId, results$Target.segments.start, results$Target.segments.end,
              praatScriptCentreOfGravity(powers=c(1.0,2.0)),
              no.progress=TRUE)

## End(Not run)
```

`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

```
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.
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

praatScriptCentreOfGravity

praatScriptIntensity

praatScriptPitch

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(segments="I"))

## Get the first 3 formants at three points during the vowel
formants <- processWithPraat(
              labbcat.url,
            results$MatchId, results$Target.segments.start, results$Target.segments.end,
              window.offset=0.5,
              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

Examples

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

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

## Get intensity for all matches
intensity <- processWithPraat(
              labbcat.url,
            results$MatchId, results$Target.segments.start, results$Target.segments.end,
              praatScriptIntensity(),
              no.progress=TRUE)

## 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
`octave.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

Examples

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

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

## Get pitch mean, max, and min for all matches
pitch <- processWithPraat(
              labbcat.url,
            results$MatchId, results$Target.segments.start, results$Target.segments.end,
              praatScriptPitch(get.mean=TRUE, get.minimum=TRUE, get.maximum=TRUE),
              no.progress=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,
  matchIds,
  startOffsets,
  endOffsets,
  praat.script,
  window.offset = 0,
  gender.attribute = "participant_gender",
  attributes = NULL,
  no.progress = FALSE
)
```

Arguments

`labbcat.url`	URL to the LaBB-CAT instance
`matchIds`	A vector of annotation IDs, e.g. the MatchId column, or the URL column, of a results set.
`startOffsets`	The start time in seconds, or a vector of start times.
`endOffsets`	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`	Optionally suppress the progress bar when multiple fragments are specified - TRUE for no progress bar.

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.

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(segments="I"))

## get F1 and F2 for the mid point of the vowel
formants <- processWithPraat(
       labbcat.url,
       results$MatchId, results$Target.segments.start, results$Target.segments.end,
       praatScriptFormants(),
       no.progress=TRUE)

## 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.segments.start, results$Target.segments.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.segments.start, results$Target.segments.end,
        readLines("acousticMeasurements.praat"))

## End(Not run)
```

Index