Package ‘tm.plugin.mail’

July 22, 2025

Title Text Mining E-Mail Plug-in

Version 0.3-1

Imports NLP (>= 0.1-2), tm (>= 0.6-1), reticulate

Description A plug-in for the tm text mining framework providing mail handling functionality.

License GPL-3

NeedsCompilation no

Author Ingo Feinerer [aut] (ORCID: <https://orcid.org/0000-0001-7656-8338>),
Wolfgang Mauerer [aut],
Kurt Hornik [aut, cre] (ORCID: <https://orcid.org/0000-0003-4198-9911>)

Maintainer Kurt Hornik <Kurt.Hornik@R-project.org>

Repository CRAN

Date/Publication 2024-09-12 13:38:44 UTC

Contents

convert_mbox_eml *Convert E-Mails From mbox Format To eml Format*

Description

Convert e-mails from mbox (i.e., several mails in a single box) format to eml (i.e., every mail in a single file) format.

Usage

```
convert_mbox_eml(mbox, dir, format = "mbox", delim = NULL)
```

Arguments

mbox	a character string or connection describing the mbox location.
dir	a character string describing the output directory.
format	see MBoxSource.
delim	see MBoxSource.

Value

No explicit return value. As a side product the directory dir contains the e-mails in eml format.

Author(s)

Ingo Feinerer and Kurt Hornik

See Also

https://www.loc.gov/preservation/digital/formats/fdd/fdd000388.shtml.

MailDocument *E-Mail Documents*

Description

Create electronic mail documents.

Usage

```
MailDocument(x,
             author = character(),
             datetimestamp = as.POSIXlt(Sys.time(), tz = "GMT"),
             description = character(),
             header = character(),
             heading = character(),
             id = character(),
             language = character(),
             origin = character(),
             ...,
             meta = NULL)
```

Arguments

`x` a character vector giving the text content.

`author` a character vector or an object of class `person` giving the author names.

`datetimestamp` an object of class `POSIXt` or a character string giving the creation date/time information. If a character string, exactly one of the ISO 8601 formats defined by `https://www.w3.org/TR/NOTE-datetime` should be used. See `parse_ISO_8601_datetime` in package **NLP** for processing such date/time information.

`description` a character string giving a description.

`header` a character vector or list giving the mail header information.

`heading` a character string giving the title or a short heading.

`id` a character string giving a unique identifier.

`language` a character string giving the language (preferably as IETF language tags, see language in package **NLP**).

`origin` a character string giving information on the source and origin.

`...` user-defined document metadata tag-value pairs.

`meta` a named list or `NULL` (default) giving all metadata. If set, all other metadata arguments are ignored.

Value

An object inheriting from `MailDocument`, `PlainTextDocument`, and `TextDocument`.

Author(s)

Ingo Feinerer and Kurt Hornik

MBoxSource *Mailbox Source*

Description

Create a mailbox source.

Usage

```
MBoxSource(mbox, format = "mbox", delim = NULL)
```

Arguments

`mbox`	a character string giving the path or URL to a mailbox stored in “mbox” format.
`format`	a character string giving the mbox format to use, with possible values `"mbox"` (default), `"mboxo"`, and `"mboxrd"`.
`delim`	a character string giving a regexp to use for finding the ‘From ’ lines delimiting the messages, or `NULL` (default), which provides suitable regexps according to the mbox format.

Details

A *mailbox source* interprets each e-mail stored in the mailbox as a document.

‘Mbox’ is a generic term for a family of related file formats used for holding collections of email messages. The messages are stored in a single mailbox text file separated by lines starting with the four characters ‘From’ followed by a space (the so-called ‘From ’ lines) and the sender’s email address.

Clearly, there will be a problem if the message bodies contain lines which also start with ‘From’ followed by a space. There are four common variants of the mbox format to deal with this problem: in *mboxo* and *mboxrd* such lines get a greater-than sign prepended, whereas in *mboxcl* and *mboxcl2* a ‘Content-Length:’ header field is used to record the message lengths. For more information, see `https://en.wikipedia.org/wiki/Mbox` and `https://www.loc.gov/preservation/digital/formats/fdd/fdd000383.shtml` which in turn points to `https://www.loc.gov/preservation/digital/formats/fdd/fdd000384.shtml` and `https://www.loc.gov/preservation/digital/formats/fdd/fdd000385.shtml` for the *mboxo* and *mboxrd* extensions.

The above LoC web page suggests that the ‘From ’ lines are always of the form ‘From *sender date moreinfo*’ where *sender* is one word without spaces or tabs and *date* (the delivery date of the message) always contains exactly 24 characters in Standard C asctime format. Thus, for the *mbox* format, the default delimiter regexp for ‘From ’ lines actually matches this form (with some timezone variants). For the *mboxo* and *mboxrd* variants, the default delimiter regexp is `"^From "`.

The `getElem()` method for class `MBoxSource` strips the prepended greater-than signs for the *mboxo* and *mboxrd* formats.

Value

An object inheriting from `MBoxSource`, `SimpleSource`, and `Source`.

Author(s)

Ingo Feinerer and Kurt Hornik

`readMail` *Read In an E-Mail Document*

Description

Return a function which reads in an electronic mail document.

Usage

```
readMail(DateFormat = character())
```

Arguments

`DateFormat`	A character vector giving date-time formats for the "Date" header field in the mail document. By default, the "basic" formats of RFC 5322 are tried.

Details

Formally this function is a function generator, i.e., it returns a function (which reads in a mail document) with a well-defined signature, but can access passed over arguments (e.g., the "Date" header format) via lexical scoping.

In version 0.3.0 of the **tm.plugin.mail** package, the reader code was switched to use the Python **email** library via CRAN package **reticulate**. Compared to previous versions, this allows to

- handle textual message bodies in character sets other than US-ASCII and the use of base64 or quoted-printable transfer encodings (RFC 2045)
- handle non-US-ASCII text data in message header fields (RFC 2047)
- correctly handle the metadata in structured header fields (RFC 5322)

For messages using the Multipurpose Internet Mail Extensions (MIME) extensions, the texts extracted from the messages are the (suitably decoded) bodies when using the 'text/plain' or 'text/html' content types, or the body parts using these types when using 'multipart/mixed' or 'multipart/alternative' (see RFC 2046 for more information). Non-MIME messages are treated like 'text/plain'. The extracted texts are represented as character vectors with length the number of extracted body parts and names giving the MIME *subtype* (`"plain"` or `"html"`).

This allows text mining applications to flexibly handle HTML content "as appropriate" by filtering on the names of the content of the `MailDocument` objects.

In case the Python processing fails or its results cannot be transferred to R (in particular, when text body parts contain embedded NULs), the reader falls back to simple header field processing appropriate for unstructered headers, and/or extracting no text. Information about problems is provided in the `problems` element of the metadata.

Value

A `function` with the following formals:

`elem` a named list with the component `content` which must hold the document to be read in.

`language` a string giving the language.

`id` a character giving a unique identifier for the created text document.

The function returns a `MailDocument` representing the text and metadata extracted from `elem$content`. The argument `id` is used as fallback if no corresponding metadata entry is found in `elem$content`.

Author(s)

Ingo Feinerer and Kurt Hornik

See Also

`Reader` for basic information on the reader infrastructure employed by package **tm**.

`strptime` for date-time format specifications.

RFC 5322, RFC 2045, RFC 2045, RFC 2047.

Examples

```
require("tm")
newsgroup <- system.file("mails", package = "tm.plugin.mail")
news <- VCorpus(DirSource(newsgroup),
                readerControl = list(reader = readMail))
inspect(news)
## Use the high-level content and metadata accessors from package 'NLP':
require("NLP")
content(news[[2]])
meta(news[[2]])
## Processed header fields of the message.
meta(news[[2]])$header
```

removeCitation *Remove E-Mail Citations*

Description

Remove citations, i.e., lines beginning with >, from an e-mail message.

Usage

```
## S3 method for class 'MailDocument'
removeCitation(x, ...)
```

Arguments

`x` A mail document.

`...` the argument `removeQuoteHeader` (default `FALSE`) giving a logical indicating if the quotation header (of the type “On *date*, *author* wrote:”) that proceeds the quoted message should be removed.

Author(s)

Ingo Feinerer

See Also

`removeMultipart` to remove non-text parts from multipart e-mail messages, and `removeSignature` to remove signature lines from e-mail messages.

Examples

```
require("tm")
newsgroup <- system.file("mails", package = "tm.plugin.mail")
news <- VCorpus(DirSource(newsgroup),
                readerControl = list(reader = readMail))
news[[8]]
removeCitation(news[[8]])
removeCitation(news[[8]], removeQuoteHeader = TRUE)
```

`removeMultipart` *Remove Non-Text Parts From E-Mails*

Description

Remove non-text parts from multipart e-mail messages.

Usage

```
## S3 method for class 'MailDocument'
removeMultipart(x, ...)
```

Arguments

`x` A mail document.

`...` Not used.

Author(s)

Ingo Feinerer

See Also

removeCitation to remove e-mail citations, and removeSignature to remove signature lines from e-mail messages.

---

removeSignature *Remove E-Mail Signatures*

Description

Remove signature lines from an e-mail message.

Usage

```
## S3 method for class 'MailDocument'
removeSignature(x, ...)
```

Arguments

`x` A mail document.

`...` the argument `marks` giving a character of signature identifications marks (in form of regular expression patterns). Note that the official signature start mark `-- ` (dash dash blank) is always considered.

Author(s)

Ingo Feinerer

See Also

removeCitation to remove e-mail citations, and removeMultipart to remove non-text parts from multipart e-mail messages.

Examples

```
require("tm")
newsgroup <- system.file("mails", package = "tm.plugin.mail")
news <- VCorpus(DirSource(newsgroup),
                readerControl = list(reader = readMail))
news[[7]]
removeSignature(news[[7]], marks = "^[+]-*[+]$")
```

threads *E-Mail Threads*

Description

Extract threads (i.e., chains of messages on a single subject) from e-mail documents.

Usage

```
threads(x)
```

Arguments

x	A corpus consisting of e-mails (MailDocuments).

Details

This function uses a one-pass algorithm for extracting the thread information by inspecting the “References” header. Some mails (e.g., reply mails appearing before their corresponding base mails) might not be tagged correctly.

Value

A list with the two named components ThreadID and ThreadDepth, listing a thread and the level of replies for each mail in the corpus x.

Examples

```
require("tm")
newsgroup <- system.file("mails", package = "tm.plugin.mail")
news <- VCorpus(DirSource(newsgroup),
                readerControl = list(reader = readMail))
vapply(news, meta, "id", FUN.VALUE = "")
lapply(news, function(x) meta(x, "header")$References)
(info <- threads(news))
lengths(split(news, info$ThreadID))
```

Index