:mod:`nntplib` --- NNTP protocol client

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 1); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 4)

Unknown directive type "module".

```
.. module:: nntplib
   :synopsis: NNTP protocol client (requires sockets).
   :deprecated:
```

Source code: :source:`Lib/nntplib.py`

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 8); *backlink*

Unknown interpreted text role "source".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 10)

Unknown directive type "index".

```
.. index::
   pair: NNTP; protocol
   single: Network News Transfer Protocol
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 14)

Unknown directive type "deprecated".

```
.. deprecated:: 3.11
   The :mod:`nntplib` module is deprecated (see :pep:`594` for details).
```

This module defines the class :class:`NNTP` which implements the client side of the Network News Transfer Protocol. It can be used to implement a news reader or poster, or automated news processors. It is compatible with RFC 3977 as well as the older RFC 977 and RFC 2980.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 19); *backlink*

Unknown interpreted text role "class".

Here are two small examples of how it can be used. To list some statistics about a newsgroup and print the subjects of the last 10 articles:

```
>>> s = nntplib.NNTP('news.gmane.io')
>>> resp, count, first, last, name = s.group('gmane.comp.python.committers')
>>> print('Group', name, 'has', count, 'articles, range', first, 'to', last)
Group gmane.comp.python.committers has 1096 articles, range 1 to 1096
>>> resp, overviews = s.over((last - 9, last))
>>> for id, over in overviews:
...     print(id, nntplib.decode_header(over['subject']))
...
1087 Re: Commit privileges for Łukasz Langa
1088 Re: 3.2 alpha 2 freeze
1089 Re: 3.2 alpha 2 freeze
1090 Re: Commit privileges for Łukasz Langa
1091 Re: Commit privileges for Łukasz Langa
1092 Updated ssh key
```

```
1093 Re: Updated ssh key
1094 Re: Updated ssh key
1095 Hello fellow committers!
1096 Re: Hello fellow committers!
>>> s.quit()
'205 Bye!'
```

To post an article from a binary file (this assumes that the article has valid headers, and that you have right to post on the particular newsgroup):

```
>>> s = nntplib.NNTP('news.gmane.io')
>>> f = open('article.txt', 'rb')
>>> s.post(f)
'240 Article posted successfully.'
>>> s.quit()
'205 Bye!'
```

The module itself defines the following classes:

Return a new :class:`NNTP` object, representing a connection to the NNTP server running on host *host*, listening at port *port*. An optional *timeout* can be specified for the socket connection. If the optional *user* and *password* are provided, or if suitable credentials are present in :file:`/.netrc` and the optional flag *usenetrc* is true, the `AUTHINFO USER` and `AUTHINFO PASS` commands are used to identify and authenticate the user to the server. If the optional flag *readermode* is true, then a `mode reader` command is sent before authentication is performed. Reader mode is sometimes necessary if you are connecting to an NNTP server on the local machine and intend to call reader-specific commands, such as `group`. If you get unexpected :exc:`NNTPPermanentError`s, you might need to set *readermode*. The :class:`NNTP` class supports the :keyword:`with` statement to unconditionally consume :exc:`OSError` exceptions and to close the NNTP connection when done, e.g.:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 63); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 63); *backlink*

Unknown interpreted text role "file".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 63); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 63); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 63); *backlink*

Unknown interpreted text role "keyword".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 63); *backlink*

Unknown interpreted text role "exc".

```
>>> from nntplib import NNTP
>>> with NNTP('news.gmane.io') as n:
...     n.group('gmane.comp.python.committers')
... # doctest: +SKIP
('211 1755 1 1755 gmane.comp.python.committers', 1755, 1, 1755, 'gmane.comp.python.committers')
>>>
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 86)

Unknown directive type "audit-event".

```
.. audit-event:: nntplib.connect self,host,port nntplib.NNTP
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 88)

Unknown directive type "audit-event".

```
.. audit-event:: nntplib.putline self,line nntplib.NNTP

   All commands will raise an :ref:`auditing event <auditing>`
   ``nntplib.putline`` with arguments ``self`` and ``line``,
   where ``line`` is the bytes about to be sent to the remote host.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 94)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.2
   *usenetrc* is now ``False`` by default.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 97)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.3
   Support for the :keyword:`with` statement was added.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 100)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.9
   If the *timeout* parameter is set to be zero, it will raise a
   :class:`ValueError` to prevent the creation of a non-blocking socket.
```

Return a new :class:`NNTP_SSL` object, representing an encrypted connection to the NNTP server running on host *host*, listening at port *port*. :class:`NNTP_SSL` objects have the same methods as :class:`NNTP` objects. If *port* is omitted, port 563 (NNTPS) is used. *ssl_context* is also optional, and is a :class:`~ssl.SSLContext` object. Please read :ref:`ssl-security` for best practices. All other parameters behave the same as for :class:`NNTP`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 106); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 106); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 106); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 106); *backlink*

Unknown interpreted text role "class".

**System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 106); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 106); *backlink*

Unknown interpreted text role "class".

Note that SSL-on-563 is discouraged per RFC 4642, in favor of STARTTLS as described below. However, some servers only support the former.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 118)

Unknown directive type "audit-event".

```
.. audit-event:: nntplib.connect self,host,port nntplib.NNTP_SSL
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 120)

Unknown directive type "audit-event".

```
.. audit-event:: nntplib.putline self,line nntplib.NNTP_SSL

   All commands will raise an :ref:`auditing event <auditing>`
   ``nntplib.putline`` with arguments ``self`` and ``line``,
   where ``line`` is the bytes about to be sent to the remote host.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 126)

Unknown directive type "versionadded".

```
.. versionadded:: 3.2
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 128)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.4
   The class now supports hostname check with
   :attr:`ssl.SSLContext.check_hostname` and *Server Name Indication* (see
   :data:`ssl.HAS_SNI`).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 133)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.9
   If the *timeout* parameter is set to be zero, it will raise a
   :class:`ValueError` to prevent the creation of a non-blocking socket.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 137)

Unknown directive type "exception".

```
.. exception:: NNTPError

   Derived from the standard exception :exc:`Exception`, this is the base
   class for all exceptions raised by the :mod:`nntplib` module.  Instances
   of this class have the following attribute:

   .. attribute:: response
```

```
        The response of the server if available, as a :class:`str` object.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 148)

Unknown directive type "exception".

```
.. exception:: NNTPReplyError

   Exception raised when an unexpected reply is received from the server.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 153)

Unknown directive type "exception".

```
.. exception:: NNTPTemporaryError

   Exception raised when a response code in the range 400--499 is received.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 158)

Unknown directive type "exception".

```
.. exception:: NNTPPermanentError

   Exception raised when a response code in the range 500--599 is received.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 163)

Unknown directive type "exception".

```
.. exception:: NNTPProtocolError

   Exception raised when a reply is received from the server that does not begin
   with a digit in the range 1--5.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 169)

Unknown directive type "exception".

```
.. exception:: NNTPDataError

   Exception raised when there is some error in the response data.
```

NNTP Objects

When connected, :class:`NNTP` and :class:`NNTP_SSL` objects support the following methods and attributes.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 179); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 179); *backlink*

Unknown interpreted text role "class".

Attributes

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 185)

Unknown directive type "attribute".

```
.. attribute:: NNTP.nntp_version

   An integer representing the version of the NNTP protocol supported by the
   server.  In practice, this should be ``2`` for servers advertising
   :rfc:`3977` compliance and ``1`` for others.

   .. versionadded:: 3.2
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 193)

Unknown directive type "attribute".

```
.. attribute:: NNTP.nntp_implementation

   A string describing the software name and version of the NNTP server,
   or :const:`None` if not advertised by the server.

   .. versionadded:: 3.2
```

Methods

The *response* that is returned as the first item in the return tuple of almost all methods is the server's response: a string beginning with a three-digit code. If the server's response indicates an error, the method raises one of the above exceptions.

Many of the following methods take an optional keyword-only argument *file*. When the *file* argument is supplied, it must be either a :term:`file object` opened for binary writing, or the name of an on-disk file to be written to. The method will then write any data returned by the server (except for the response line and the terminating dot) to the file; any list of lines, tuples or objects that the method normally returns will be empty.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 208); *backlink*

Unknown interpreted text role "term".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 215)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.2
   Many of the following methods have been reworked and fixed, which makes
   them incompatible with their 3.1 counterparts.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 220)

Unknown directive type "method".

```
.. method:: NNTP.quit()

   Send a ``QUIT`` command and close the connection.  Once this method has been
   called, no other methods of the NNTP object should be called.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 226)

Unknown directive type "method".

```
.. method:: NNTP.getwelcome()

   Return the welcome message sent by the server in reply to the initial
   connection.  (This message sometimes contains disclaimers or help information
   that may be relevant to the user.)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 233)

Unknown directive type "method".

```
.. method:: NNTP.getcapabilities()

   Return the :rfc:`3977` capabilities advertised by the server, as a
   :class:`dict` instance mapping capability names to (possibly empty) lists
   of values. On legacy servers which don't understand the ``CAPABILITIES``
   command, an empty dictionary is returned instead.

      >>> s = NNTP('news.gmane.io')
      >>> 'POST' in s.getcapabilities()
      True

   .. versionadded:: 3.2
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 247)

Unknown directive type "method".

```
.. method:: NNTP.login(user=None, password=None, usenetrc=True)

   Send ``AUTHINFO`` commands with the user name and password.  If *user*
   and *password* are ``None`` and *usenetrc* is true, credentials from
   ``~/.netrc`` will be used if possible.

   Unless intentionally delayed, login is normally performed during the
   :class:`NNTP` object initialization and separately calling this function
   is unnecessary.  To force authentication to be delayed, you must not set
   *user* or *password* when creating the object, and must set *usenetrc* to
   False.

   .. versionadded:: 3.2
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 262)

Unknown directive type "method".

```
.. method:: NNTP.starttls(context=None)

   Send a ``STARTTLS`` command.  This will enable encryption on the NNTP
   connection.  The *context* argument is optional and should be a
   :class:`ssl.SSLContext` object.  Please read :ref:`ssl-security` for best
   practices.

   Note that this may not be done after authentication information has
   been transmitted, and authentication occurs by default if possible during a
   :class:`NNTP` object initialization.  See :meth:`NNTP.login` for information
   on suppressing this behavior.

   .. versionadded:: 3.2

   .. versionchanged:: 3.4
      The method now supports hostname check with
      :attr:`ssl.SSLContext.check_hostname` and *Server Name Indication* (see
      :data:`ssl.HAS_SNI`).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 281)

Unknown directive type "method".

```
.. method:: NNTP.newgroups(date, *, file=None)

   Send a ``NEWGROUPS`` command.  The *date* argument should be a
   :class:`datetime.date` or :class:`datetime.datetime` object.
   Return a pair ``(response, groups)`` where *groups* is a list representing
   the groups that are new since the given *date*. If *file* is supplied,
   though, then *groups* will be empty.

      >>> from datetime import date, timedelta
      >>> resp, groups = s.newgroups(date.today() - timedelta(days=3))
      >>> len(groups) # doctest: +SKIP
      85
      >>> groups[0] # doctest: +SKIP
      GroupInfo(group='gmane.network.tor.devel', last='4', first='1', flag='m')
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 297)

Unknown directive type "method".

```
.. method:: NNTP.newnews(group, date, *, file=None)

   Send a ``NEWNEWS`` command.  Here, *group* is a group name or ``'*'``, and
   *date* has the same meaning as for :meth:`newgroups`.  Return a pair
   ``(response, articles)`` where *articles* is a list of message ids.

   This command is frequently disabled by NNTP server administrators.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 306)

Unknown directive type "method".

```
.. method:: NNTP.list(group_pattern=None, *, file=None)

   Send a ``LIST`` or ``LIST ACTIVE`` command.  Return a pair
   ``(response, list)`` where *list* is a list of tuples representing all
   the groups available from this NNTP server, optionally matching the
   pattern string *group_pattern*.  Each tuple has the form
   ``(group, last, first, flag)``, where *group* is a group name, *last*
   and *first* are the last and first article numbers, and *flag* usually
   takes one of these values:

   * ``y``: Local postings and articles from peers are allowed.
   * ``m``: The group is moderated and all postings must be approved.
   * ``n``: No local postings are allowed, only articles from peers.
   * ``j``: Articles from peers are filed in the junk group instead.
   * ``x``: No local postings, and articles from peers are ignored.
   * ``=foo.bar``: Articles are filed in the ``foo.bar`` group instead.

   If *flag* has another value, then the status of the newsgroup should be
   considered unknown.

   This command can return very large results, especially if *group_pattern*
   is not specified.  It is best to cache the results offline unless you
   really need to refresh them.

   .. versionchanged:: 3.2
      *group_pattern* was added.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 334)

Unknown directive type "method".

```
.. method:: NNTP.descriptions(grouppattern)

   Send a ``LIST NEWSGROUPS`` command, where *grouppattern* is a wildmat string as
   specified in :rfc:`3977` (it's essentially the same as DOS or UNIX shell wildcard
   strings).  Return a pair ``(response, descriptions)``, where *descriptions*
```

```
   is a dictionary mapping group names to textual descriptions.

      >>> resp, descs = s.descriptions('gmane.comp.python.*')
      >>> len(descs) # doctest: +SKIP
      295
      >>> descs.popitem() # doctest: +SKIP
      ('gmane.comp.python.bio.general', 'BioPython discussion list (Moderated)')
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 348)

Unknown directive type "method".

```
.. method:: NNTP.description(group)

   Get a description for a single group *group*.  If more than one group matches
   (if 'group' is a real wildmat string), return the first match.   If no group
   matches, return an empty string.

   This elides the response code from the server.  If the response code is needed,
   use :meth:`descriptions`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 358)

Unknown directive type "method".

```
.. method:: NNTP.group(name)

   Send a ``GROUP`` command, where *name* is the group name.  The group is
   selected as the current group, if it exists.  Return a tuple
   ``(response, count, first, last, name)`` where *count* is the (estimated)
   number of articles in the group, *first* is the first article number in
   the group, *last* is the last article number in the group, and *name*
   is the group name.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 368)

Unknown directive type "method".

```
.. method:: NNTP.over(message_spec, *, file=None)

   Send an ``OVER`` command, or an ``XOVER`` command on legacy servers.
   *message_spec* can be either a string representing a message id, or
   a ``(first, last)`` tuple of numbers indicating a range of articles in
   the current group, or a ``(first, None)`` tuple indicating a range of
   articles starting from *first* to the last article in the current group,
   or :const:`None` to select the current article in the current group.

   Return a pair ``(response, overviews)``.  *overviews* is a list of
   ``(article_number, overview)`` tuples, one for each article selected
   by *message_spec*.  Each *overview* is a dictionary with the same number
   of items, but this number depends on the server.  These items are either
   message headers (the key is then the lower-cased header name) or metadata
   items (the key is then the metadata name prepended with ``":"``).  The
   following items are guaranteed to be present by the NNTP specification:

   * the ``subject``, ``from``, ``date``, ``message-id`` and ``references``
     headers
   * the ``:bytes`` metadata: the number of bytes in the entire raw article
     (including headers and body)
   * the ``:lines`` metadata: the number of lines in the article body

   The value of each item is either a string, or :const:`None` if not present.

   It is advisable to use the :func:`decode_header` function on header
   values when they may contain non-ASCII characters::

      >>> _, _, first, last, _ = s.group('gmane.comp.python.devel')
      >>> resp, overviews = s.over((last, last))
      >>> art_num, over = overviews[0]
```

```
      >>> art_num
      117216
      >>> list(over.keys())
      ['xref', 'from', ':lines', ':bytes', 'references', 'date', 'message-id', 'subject']
      >>> over['from']
      '=?UTF-8?B?Ik1hcnRpbiB2LiBMw7Z3aXMi?= <martin@v.loewis.de>'
      >>> nntplib.decode_header(over['from'])
      '"Martin v. Löwis" <martin@v.loewis.de>'

   .. versionadded:: 3.2
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 411)

Unknown directive type "method".

```
.. method:: NNTP.help(*, file=None)

   Send a ``HELP`` command.  Return a pair ``(response, list)`` where *list* is a
   list of help strings.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 417)

Unknown directive type "method".

```
.. method:: NNTP.stat(message_spec=None)

   Send a ``STAT`` command, where *message_spec* is either a message id
   (enclosed in ``'<'`` and ``'>'``) or an article number in the current group.
   If *message_spec* is omitted or :const:`None`, the current article in the
   current group is considered.  Return a triple ``(response, number, id)``
   where *number* is the article number and *id* is the message id.

      >>> _, _, first, last, _ = s.group('gmane.comp.python.devel')
      >>> resp, number, message_id = s.stat(first)
      >>> number, message_id
      (9099, '<20030112190404.GE29873@epoch.metaslash.com>')
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 431)

Unknown directive type "method".

```
.. method:: NNTP.next()

   Send a ``NEXT`` command.  Return as for :meth:`.stat`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 436)

Unknown directive type "method".

```
.. method:: NNTP.last()

   Send a ``LAST`` command.  Return as for :meth:`.stat`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 441)

Unknown directive type "method".

```
.. method:: NNTP.article(message_spec=None, *, file=None)

   Send an ``ARTICLE`` command, where *message_spec* has the same meaning as
   for :meth:`.stat`.  Return a tuple ``(response, info)`` where *info*
```

```
is a :class:`~collections.namedtuple` with three attributes *number*,
*message_id* and *lines* (in that order).  *number* is the article number
in the group (or 0 if the information is not available), *message_id* the
message id as a string, and *lines* a list of lines (without terminating
newlines) comprising the raw message including headers and body.

   >>> resp, info = s.article('<20030112190404.GE29873@epoch.metaslash.com>')
   >>> info.number
   0
   >>> info.message_id
   '<20030112190404.GE29873@epoch.metaslash.com>'
   >>> len(info.lines)
   65
   >>> info.lines[0]
   b'Path: main.gmane.org!not-for-mail'
   >>> info.lines[1]
   b'From: Neal Norwitz <neal@metaslash.com>'
   >>> info.lines[-3:]
   [b'There is a patch for 2.3 as well as 2.2.', b'', b'Neal']
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 466)

Unknown directive type "method".

```
.. method:: NNTP.head(message_spec=None, *, file=None)

   Same as :meth:`article()`, but sends a ``HEAD`` command.  The *lines*
   returned (or written to *file*) will only contain the message headers, not
   the body.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 473)

Unknown directive type "method".

```
.. method:: NNTP.body(message_spec=None, *, file=None)

   Same as :meth:`article()`, but sends a ``BODY`` command.  The *lines*
   returned (or written to *file*) will only contain the message body, not the
   headers.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 480)

Unknown directive type "method".

```
.. method:: NNTP.post(data)

   Post an article using the ``POST`` command.  The *data* argument is either
   a :term:`file object` opened for binary reading, or any iterable of bytes
   objects (representing raw lines of the article to be posted).  It should
   represent a well-formed news article, including the required headers.  The
   :meth:`post` method automatically escapes lines beginning with ``.`` and
   appends the termination line.

   If the method succeeds, the server's response is returned.  If the server
   refuses posting, a :class:`NNTPReplyError` is raised.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 493)

Unknown directive type "method".

```
.. method:: NNTP.ihave(message_id, data)

   Send an ``IHAVE`` command. *message_id* is the id of the message to send
   to the server (enclosed in  ``'<'`` and ``'>'``).  The *data* parameter
   and the return value are the same as for :meth:`post()`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 500)

Unknown directive type "method".

```
.. method:: NNTP.date()

   Return a pair ``(response, date)``.  *date* is a :class:`~datetime.datetime`
   object containing the current date and time of the server.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 506)

Unknown directive type "method".

```
.. method:: NNTP.slave()

   Send a ``SLAVE`` command.  Return the server's *response*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 511)

Unknown directive type "method".

```
.. method:: NNTP.set_debuglevel(level)

   Set the instance's debugging level.  This controls the amount of debugging
   output printed.  The default, ``0``, produces no debugging output.  A value of
   ``1`` produces a moderate amount of debugging output, generally a single line
   per request or response.  A value of ``2`` or higher produces the maximum amount
   of debugging output, logging each line sent and received on the connection
   (including message text).
```

The following are optional NNTP extensions defined in RFC 2980. Some of them have been superseded by newer commands in RFC 3977.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 525)

Unknown directive type "method".

```
.. method:: NNTP.xhdr(hdr, str, *, file=None)

   Send an ``XHDR`` command.  The *hdr* argument is a header keyword, e.g.
   ``'subject'``.  The *str* argument should have the form ``'first-last'``
   where *first* and *last* are the first and last article numbers to search.
   Return a pair ``(response, list)``, where *list* is a list of pairs ``(id,
   text)``, where *id* is an article number (as a string) and *text* is the text of
   the requested header for that article. If the *file* parameter is supplied, then
   the output of the  ``XHDR`` command is stored in a file.  If *file* is a string,
   then the method will open a file with that name, write to it  then close it.
   If *file* is a :term:`file object`, then it will start calling :meth:`write` on
   it to store the lines of the command output. If *file* is supplied, then the
   returned *list* is an empty list.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 540)

Unknown directive type "method".

```
.. method:: NNTP.xover(start, end, *, file=None)

   Send an ``XOVER`` command.  *start* and *end* are article numbers
   delimiting the range of articles to select.  The return value is the
   same of for :meth:`over()`.  It is recommended to use :meth:`over()`
```

```
instead, since it will automatically use the newer ``OVER`` command
if available.
```

Utility functions

The module also defines the following utility function:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]nntplib.rst, line 555)

Unknown directive type "function".

```
.. function:: decode_header(header_str)

   Decode a header value, un-escaping any escaped non-ASCII characters.
   *header_str* must be a :class:`str` object.  The unescaped value is
   returned.  Using this function is recommended to display some headers
   in a human readable form::

      >>> decode_header("Some subject")
      'Some subject'
      >>> decode_header("=?ISO-8859-15?Q?D=E9buter_en_Python?=")
      'Débuter en Python'
      >>> decode_header("Re: =?UTF-8?B?cHJvYmzDqG1lIGRlIG1hdHJpY2U=?=")
      'Re: problème de matrice'
```