:mod:`http.client` --- HTTP protocol client

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

Unknown directive type "module".

```
.. module:: http.client
   :synopsis: HTTP and HTTPS protocol client (requires sockets).
```

Source code: :source:`Lib/http/client.py`

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 7); *backlink*

Unknown interpreted text role "source".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 9)

Unknown directive type "index".

```
.. index::
   pair: HTTP; protocol
   single: HTTP; http.client (standard module)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 13)

Unknown directive type "index".

```
.. index:: module: urllib.request
```

This module defines classes which implement the client side of the HTTP and HTTPS protocols. It is normally not used directly --- the module :mod:`urllib.request` uses it to handle URLs that use HTTP and HTTPS.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 17); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 21)

Unknown directive type "seealso".

```
.. seealso::

    The `Requests package <https://requests.readthedocs.io/en/master/>`_
    is recommended for a higher-level HTTP client interface.
```

Note

HTTPS support is only available if Python was compiled with SSL support (through the :mod:`ssl` module).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 28); *backlink*

Unknown interpreted text role "mod".

The module provides the following classes:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 34)

Invalid class attribute value for "class" directive: "HTTPConnection(host, port=None[, timeout], source_address=None, \ blocksize=8192)".

```
.. class:: HTTPConnection(host, port=None[, timeout], source_address=None, \
                          blocksize=8192)

   An :class:`HTTPConnection` instance represents one transaction with an HTTP
   server.  It should be instantiated passing it a host and optional port
   number.  If no port number is passed, the port is extracted from the host
   string if it has the form ``host:port``, else the default HTTP port (80) is
   used.  If the optional *timeout* parameter is given, blocking
   operations (like connection attempts) will timeout after that many seconds
   (if it is not given, the global default timeout setting is used).
   The optional *source_address* parameter may be a tuple of a (host, port)
   to use as the source address the HTTP connection is made from.
   The optional *blocksize* parameter sets the buffer size in bytes for
   sending a file-like message body.

   For example, the following calls all create instances that connect to the server
   at the same host and port::

      >>> h1 = http.client.HTTPConnection('www.python.org')
      >>> h2 = http.client.HTTPConnection('www.python.org:80')
      >>> h3 = http.client.HTTPConnection('www.python.org', 80)
      >>> h4 = http.client.HTTPConnection('www.python.org', 80, timeout=10)

   .. versionchanged:: 3.2
      *source_address* was added.

   .. versionchanged:: 3.4
      The  *strict* parameter was removed. HTTP 0.9-style "Simple Responses" are
      not longer supported.

   .. versionchanged:: 3.7
      *blocksize* parameter was added.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 68)

Invalid class attribute value for "class" directive: "HTTPSConnection(host, port=None, key_file=None, \ cert_file=None[, timeout], \ source_address=None, *, context=None, \ check_hostname=None, blocksize=8192)".

```
.. class:: HTTPSConnection(host, port=None, key_file=None, \
                           cert_file=None[, timeout], \
                           source_address=None, *, context=None, \
                           check_hostname=None, blocksize=8192)

   A subclass of :class:`HTTPConnection` that uses SSL for communication with
   secure servers.  Default port is ``443``.  If *context* is specified, it
   must be a :class:`ssl.SSLContext` instance describing the various SSL
   options.

   Please read :ref:`ssl-security` for more information on best practices.

   .. versionchanged:: 3.2
      *source_address*, *context* and *check_hostname* were added.

   .. versionchanged:: 3.2
      This class now supports HTTPS virtual hosts if possible (that is,
      if :data:`ssl.HAS_SNI` is true).

   .. versionchanged:: 3.4
      The *strict* parameter was removed. HTTP 0.9-style "Simple Responses" are
      no longer supported.

   .. versionchanged:: 3.4.3
      This class now performs all the necessary certificate and hostname checks
```

```
   by default. To revert to the previous, unverified, behavior
   :func:`ssl._create_unverified_context` can be passed to the *context*
   parameter.

.. versionchanged:: 3.8
   This class now enables TLS 1.3
   :attr:`ssl.SSLContext.post_handshake_auth` for the default *context* or
   when *cert_file* is passed with a custom *context*.

.. versionchanged:: 3.10
   This class now sends an ALPN extension with protocol indicator
   ``http/1.1`` when no *context* is given. Custom *context* should set
   ALPN protocols with :meth:`~ssl.SSLContext.set_alpn_protocol`.

.. deprecated:: 3.6

    *key_file* and *cert_file* are deprecated in favor of *context*.
    Please use :meth:`ssl.SSLContext.load_cert_chain` instead, or let
    :func:`ssl.create_default_context` select the system's trusted CA
    certificates for you.

    The *check_hostname* parameter is also deprecated; the
    :attr:`ssl.SSLContext.check_hostname` attribute of *context* should
    be used instead.
```

Class whose instances are returned upon successful connection. Not instantiated directly by user.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 124)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.4
   The *strict* parameter was removed. HTTP 0.9 style "Simple Responses" are
   no longer supported.
```

This module provides the following function:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 130)

Unknown directive type "function".

```
.. function:: parse_headers(fp)

   Parse the headers from a file pointer *fp* representing a HTTP
   request/response. The file has to be a :class:`BufferedIOBase` reader
   (i.e. not text) and must provide a valid :rfc:`2822` style header.

   This function returns an instance of :class:`http.client.HTTPMessage`
   that holds the header fields, but no payload
   (the same as :attr:`HTTPResponse.msg`
   and :attr:`http.server.BaseHTTPRequestHandler.headers`).
   After returning, the file pointer *fp* is ready to read the HTTP body.

   .. note::
      :meth:`parse_headers` does not parse the start-line of a HTTP message;
      it only parses the ``Name: value`` lines. The file has to be ready to
      read these field lines, so the first line should already be consumed
      before calling the function.
```

The following exceptions are raised as appropriate:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 151)

Unknown directive type "exception".

```
.. exception:: HTTPException

   The base class of the other exceptions in this module.  It is a subclass of
   :exc:`Exception`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 157)

Unknown directive type "exception".

```
.. exception:: NotConnected

   A subclass of :exc:`HTTPException`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 162)

Unknown directive type "exception".

```
.. exception:: InvalidURL

   A subclass of :exc:`HTTPException`, raised if a port is given and is either
   non-numeric or empty.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 168)

Unknown directive type "exception".

```
.. exception:: UnknownProtocol

   A subclass of :exc:`HTTPException`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 173)

Unknown directive type "exception".

```
.. exception:: UnknownTransferEncoding

   A subclass of :exc:`HTTPException`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 178)

Unknown directive type "exception".

```
.. exception:: UnimplementedFileMode

   A subclass of :exc:`HTTPException`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 183)

Unknown directive type "exception".

```
.. exception:: IncompleteRead

   A subclass of :exc:`HTTPException`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 188)

Unknown directive type "exception".

```
.. exception:: ImproperConnectionState
```

```
   A subclass of :exc:`HTTPException`.
```

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

Unknown directive type "exception".

```
.. exception:: CannotSendRequest

   A subclass of :exc:`ImproperConnectionState`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 198)

Unknown directive type "exception".

```
.. exception:: CannotSendHeader

   A subclass of :exc:`ImproperConnectionState`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 203)

Unknown directive type "exception".

```
.. exception:: ResponseNotReady

   A subclass of :exc:`ImproperConnectionState`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 208)

Unknown directive type "exception".

```
.. exception:: BadStatusLine

   A subclass of :exc:`HTTPException`.  Raised if a server responds with a HTTP
   status code that we don't understand.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 214)

Unknown directive type "exception".

```
.. exception:: LineTooLong

   A subclass of :exc:`HTTPException`.  Raised if an excessively long line
   is received in the HTTP protocol from the server.
```

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

Unknown directive type "exception".

```
.. exception:: RemoteDisconnected

   A subclass of :exc:`ConnectionResetError` and :exc:`BadStatusLine`.  Raised
   by :meth:`HTTPConnection.getresponse` when the attempt to read the response
   results in no data read from the connection, indicating that the remote end
   has closed the connection.

   .. versionadded:: 3.5
```

```
Previously, :exc:`BadStatusLine`\ ``('')`` was raised.
```

The constants defined in this module are:

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

Unknown directive type "data".

```
.. data:: HTTP_PORT

   The default port for the HTTP protocol (always ``80``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 237)

Unknown directive type "data".

```
.. data:: HTTPS_PORT

   The default port for the HTTPS protocol (always ``443``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 241)

Unknown directive type "data".

```
.. data:: responses

   This dictionary maps the HTTP 1.1 status codes to the W3C names.

   Example: ``http.client.responses[http.client.NOT_FOUND]`` is ``'Not Found'``.
```

See :ref:`http-status-codes` for a list of HTTP status codes that are available in this module as constants.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 247); *backlink*

Unknown interpreted text role "ref".

HTTPConnection Objects

:class:`HTTPConnection` instances have the following methods:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 256); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 259)

Unknown directive type "method".

```
.. method:: HTTPConnection.request(method, url, body=None, headers={}, *, \
            encode_chunked=False)

   This will send a request to the server using the HTTP request
   method *method* and the selector *url*.

   If *body* is specified, the specified data is sent after the headers are
   finished.  It may be a :class:`str`, a :term:`bytes-like object`, an
   open :term:`file object`, or an iterable of :class:`bytes`.  If *body*
   is a string, it is encoded as ISO-8859-1, the default for HTTP.  If it
   is a bytes-like object, the bytes are sent as is.  If it is a :term:`file
   object`, the contents of the file is sent; this file object should
   support at least the ``read()`` method.  If the file object is an
   instance of :class:`io.TextIOBase`, the data returned by the ``read()``
```

```
   method will be encoded as ISO-8859-1, otherwise the data returned by
   ``read()`` is sent as is.  If *body* is an iterable, the elements of the
   iterable are sent as is until the iterable is exhausted.

   The *headers* argument should be a mapping of extra HTTP headers to send
   with the request.

   If *headers* contains neither Content-Length nor Transfer-Encoding,
   but there is a request body, one of those
   header fields will be added automatically.  If
   *body* is ``None``, the Content-Length header is set to ``0`` for
   methods that expect a body (``PUT``, ``POST``, and ``PATCH``).  If
   *body* is a string or a bytes-like object that is not also a
   :term:`file <file object>`, the Content-Length header is
   set to its length.  Any other type of *body* (files
   and iterables in general) will be chunk-encoded, and the
   Transfer-Encoding header will automatically be set instead of
   Content-Length.

   The *encode_chunked* argument is only relevant if Transfer-Encoding is
   specified in *headers*.  If *encode_chunked* is ``False``, the
   HTTPConnection object assumes that all encoding is handled by the
   calling code.  If it is ``True``, the body will be chunk-encoded.

   .. note::
      Chunked transfer encoding has been added to the HTTP protocol
      version 1.1.  Unless the HTTP server is known to handle HTTP 1.1,
      the caller must either specify the Content-Length, or must pass a
      :class:`str` or bytes-like object that is not also a file as the
      body representation.

   .. versionadded:: 3.2
      *body* can now be an iterable.

   .. versionchanged:: 3.6
      If neither Content-Length nor Transfer-Encoding are set in
      *headers*, file and iterable *body* objects are now chunk-encoded.
      The *encode_chunked* argument was added.
      No attempt is made to determine the Content-Length for file
      objects.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 314)

Unknown directive type "method".

```
.. method:: HTTPConnection.getresponse()

   Should be called after a request is sent to get the response from the server.
   Returns an :class:`HTTPResponse` instance.

   .. note::

      Note that you must have read the whole response before you can send a new
      request to the server.

   .. versionchanged:: 3.5
      If a :exc:`ConnectionError` or subclass is raised, the
      :class:`HTTPConnection` object will be ready to reconnect when
      a new request is sent.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 330)

Unknown directive type "method".

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

   Set the debugging level.  The default debug level is ``0``, meaning no
   debugging output is printed.  Any value greater than ``0`` will cause all
   currently defined debug output to be printed to stdout.  The ``debuglevel``
   is passed to any new :class:`HTTPResponse` objects that are created.

   .. versionadded:: 3.1
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 340)

Unknown directive type "method".

```
.. method:: HTTPConnection.set_tunnel(host, port=None, headers=None)

   Set the host and the port for HTTP Connect Tunnelling. This allows running
   the connection through a proxy server.

   The host and port arguments specify the endpoint of the tunneled connection
   (i.e. the address included in the CONNECT request, *not* the address of the
   proxy server).

   The headers argument should be a mapping of extra HTTP headers to send with
   the CONNECT request.

   For example, to tunnel through a HTTPS proxy server running locally on port
   8080, we would pass the address of the proxy to the :class:`HTTPSConnection`
   constructor, and the address of the host that we eventually want to reach to
   the :meth:`~HTTPConnection.set_tunnel` method::

      >>> import http.client
      >>> conn = http.client.HTTPSConnection("localhost", 8080)
      >>> conn.set_tunnel("www.python.org")
      >>> conn.request("HEAD","/index.html")

   .. versionadded:: 3.2
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 365)

Unknown directive type "method".

```
.. method:: HTTPConnection.connect()

   Connect to the server specified when the object was created.  By default,
   this is called automatically when making a request if the client does not
   already have a connection.

   .. audit-event:: http.client.connect self,host,port http.client.HTTPConnection.connect
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 374)

Unknown directive type "method".

```
.. method:: HTTPConnection.close()

   Close the connection to the server.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 379)

Unknown directive type "attribute".

```
.. attribute:: HTTPConnection.blocksize

   Buffer size in bytes for sending a file-like message body.

   .. versionadded:: 3.7
```

As an alternative to using the :meth:`request` method described above, you can also send your request step by step, by using the four functions below.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 386); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 390)

Unknown directive type "method".

```
.. method:: HTTPConnection.putrequest(method, url, skip_host=False, \
                                      skip_accept_encoding=False)

   This should be the first call after the connection to the server has been
   made. It sends a line to the server consisting of the *method* string,
   the *url* string, and the HTTP version (``HTTP/1.1``).  To disable automatic
   sending of ``Host:`` or ``Accept-Encoding:`` headers (for example to accept
   additional content encodings), specify *skip_host* or *skip_accept_encoding*
   with non-False values.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 401)

Unknown directive type "method".

```
.. method:: HTTPConnection.putheader(header, argument[, ...])

   Send an :rfc:`822`\ -style header to the server.  It sends a line to the server
   consisting of the header, a colon and a space, and the first argument.  If more
   arguments are given, continuation lines are sent, each consisting of a tab and
   an argument.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 409)

Unknown directive type "method".

```
.. method:: HTTPConnection.endheaders(message_body=None, *, encode_chunked=False)

   Send a blank line to the server, signalling the end of the headers. The
   optional *message_body* argument can be used to pass a message body
   associated with the request.

   If *encode_chunked* is ``True``, the result of each iteration of
   *message_body* will be chunk-encoded as specified in :rfc:`7230`,
   Section 3.3.1.  How the data is encoded is dependent on the type of
   *message_body*.  If *message_body* implements the :ref:`buffer interface
   <bufferobjects>` the encoding will result in a single chunk.
   If *message_body* is a :class:`collections.abc.Iterable`, each iteration
   of *message_body* will result in a chunk.  If *message_body* is a
   :term:`file object`, each call to ``.read()`` will result in a chunk.
   The method automatically signals the end of the chunk-encoded data
   immediately after *message_body*.

   .. note:: Due to the chunked encoding specification, empty chunks
      yielded by an iterator body will be ignored by the chunk-encoder.
      This is to avoid premature termination of the read of the request by
      the target server due to malformed encoding.

   .. versionadded:: 3.6
      Chunked encoding support.  The *encode_chunked* parameter was
      added.
```

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

Unknown directive type "method".

```
.. method:: HTTPConnection.send(data)

   Send data to the server.  This should be used directly only after the
   :meth:`endheaders` method has been called and before :meth:`getresponse` is
   called.
```

```
.. audit-event:: http.client.send self,data http.client.HTTPConnection.send
```

HTTPResponse Objects

An :class:`HTTPResponse` instance wraps the HTTP response from the server. It provides access to the request headers and the entity body. The response is an iterable object and can be used in a with statement.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 450); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 455)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.5
   The :class:`io.BufferedIOBase` interface is now implemented and
   all of its reader operations are supported.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 460)

Unknown directive type "method".

```
.. method:: HTTPResponse.read([amt])

   Reads and returns the response body, or up to the next *amt* bytes.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 464)

Unknown directive type "method".

```
.. method:: HTTPResponse.readinto(b)

   Reads up to the next len(b) bytes of the response body into the buffer *b*.
   Returns the number of bytes read.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 471)

Unknown directive type "method".

```
.. method:: HTTPResponse.getheader(name, default=None)

   Return the value of the header *name*, or *default* if there is no header
   matching *name*.  If there is more than one  header with the name *name*,
   return all of the values joined by ', '.  If 'default' is any iterable other
   than a single string, its elements are similarly returned joined by commas.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 478)

Unknown directive type "method".

```
.. method:: HTTPResponse.getheaders()

   Return a list of (header, value) tuples.
```

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

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 482)

Unknown directive type "method".

```
.. method:: HTTPResponse.fileno()

   Return the ``fileno`` of the underlying socket.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 486)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.msg

   A :class:`http.client.HTTPMessage` instance containing the response
   headers.  :class:`http.client.HTTPMessage` is a subclass of
   :class:`email.message.Message`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 492)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.version

   HTTP protocol version used by server.  10 for HTTP/1.0, 11 for HTTP/1.1.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 496)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.url

   URL of the resource retrieved, commonly used to determine if a redirect was followed.
```

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

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.headers

   Headers of the response in the form of an :class:`email.message.EmailMessage` instance.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 504)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.status

   Status code returned by server.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 508)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.reason

   Reason phrase returned by server.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 512)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.debuglevel

   A debugging hook.  If :attr:`debuglevel` is greater than zero, messages
   will be printed to stdout as the response is read and parsed.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 517)

Unknown directive type "attribute".

```
.. attribute:: HTTPResponse.closed

   Is ``True`` if the stream is closed.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 521)

Unknown directive type "method".

```
.. method:: HTTPResponse.geturl()

   .. deprecated:: 3.9
      Deprecated in favor of :attr:`~HTTPResponse.url`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 526)

Unknown directive type "method".

```
.. method:: HTTPResponse.info()

   .. deprecated:: 3.9
      Deprecated in favor of :attr:`~HTTPResponse.headers`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst, line 531)

Unknown directive type "method".

```
.. method:: HTTPResponse.getstatus()

   .. deprecated:: 3.9
      Deprecated in favor of :attr:`~HTTPResponse.status`.
```

Examples

Here is an example session that uses the `GET` method:

```
>>> import http.client
>>> conn = http.client.HTTPSConnection("www.python.org")
>>> conn.request("GET", "/")
>>> r1 = conn.getresponse()
>>> print(r1.status, r1.reason)
200 OK
>>> data1 = r1.read()  # This will return entire content.
>>> # The following example demonstrates reading data in chunks.
>>> conn.request("GET", "/")
>>> r1 = conn.getresponse()
>>> while chunk := r1.read(200):
...     print(repr(chunk))
b'<!doctype html>\n<!--[if"...
...
>>> # Example of an invalid request
>>> conn = http.client.HTTPSConnection("docs.python.org")
>>> conn.request("GET", "/parrot.spam")
>>> r2 = conn.getresponse()
>>> print(r2.status, r2.reason)
404 Not Found
>>> data2 = r2.read()
```

```
>>> conn.close()
```

Here is an example session that uses the `HEAD` method. Note that the `HEAD` method never returns any data.

```
>>> import http.client
>>> conn = http.client.HTTPSConnection("www.python.org")
>>> conn.request("HEAD", "/")
>>> res = conn.getresponse()
>>> print(res.status, res.reason)
200 OK
>>> data = res.read()
>>> print(len(data))
0
>>> data == b''
True
```

Here is an example session that shows how to `POST` requests:

```
>>> import http.client, urllib.parse
>>> params = urllib.parse.urlencode({'@number': 12524, '@type': 'issue', '@action': 'show'})
>>> headers = {"Content-type": "application/x-www-form-urlencoded",
...            "Accept": "text/plain"}
>>> conn = http.client.HTTPConnection("bugs.python.org")
>>> conn.request("POST", "", params, headers)
>>> response = conn.getresponse()
>>> print(response.status, response.reason)
302 Found
>>> data = response.read()
>>> data
b'Redirecting to <a href="http://bugs.python.org/issue12524">http://bugs.python.org/issue12524</a>'
>>> conn.close()
```

Client side `HTTP PUT` requests are very similar to `POST` requests. The difference lies only the server side where HTTP server will allow resources to be created via `PUT` request. It should be noted that custom HTTP methods are also handled in :class:`urllib.request.Request` by setting the appropriate method attribute. Here is an example session that shows how to send a `PUT` request using http.client:

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst`, line 595); *backlink*
>
> Unknown interpreted text role "class".

```
>>> # This creates an HTTP message
>>> # with the content of BODY as the enclosed representation
>>> # for the resource http://localhost:8080/file
...
>>> import http.client
>>> BODY = "***filecontents***"
>>> conn = http.client.HTTPConnection("localhost", 8080)
>>> conn.request("PUT", "/file", BODY)
>>> response = conn.getresponse()
>>> print(response.status, response.reason)
200, OK
```

HTTPMessage Objects

An :class:`http.client.HTTPMessage` instance holds the headers from an HTTP response. It is implemented using the :class:`email.message.Message` class.

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst`, line 619); *backlink*
>
> Unknown interpreted text role "class".

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\(cpython-main)(Doc)(library)http.client.rst`, line 619); *backlink*
>
> Unknown interpreted text role "class".