:mod:`wsgiref` --- WSGI Utilities and Reference Implementation

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

Unknown directive type "module".

```
.. module:: wsgiref
   :synopsis: WSGI Utilities and Reference Implementation.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 7)

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Phillip J. Eby <pje@telecommunity.com>
```

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

Unknown directive type "sectionauthor".

```
.. sectionauthor:: Phillip J. Eby <pje@telecommunity.com>
```

The Web Server Gateway Interface (WSGI) is a standard interface between web server software and web applications written in Python. Having a standard interface makes it easy to use an application that supports WSGI with a number of different web servers.

Only authors of web servers and programming frameworks need to know every detail and corner case of the WSGI design. You don't need to understand every detail of WSGI just to install a WSGI application or to write a web application using an existing framework.

:mod:`wsgiref` is a reference implementation of the WSGI specification that can be used to add WSGI support to a web server or framework. It provides utilities for manipulating WSGI environment variables and response headers, base classes for implementing WSGI servers, a demo HTTP server that serves WSGI applications, and a validation tool that checks WSGI servers and applications for conformance to the WSGI specification (PEP 3333).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 22); *backlink*

Unknown interpreted text role "mod".

See wsgi.readthedocs.io for more information about WSGI, and links to tutorials and other resources.

:mod:`wsgiref.util` -- WSGI environment utilities

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 35); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 38)

Unknown directive type "module".

```
.. module:: wsgiref.util
   :synopsis: WSGI environment utilities.
```

This module provides a variety of utility functions for working with WSGI environments. A WSGI environment is a dictionary containing HTTP request variables as described in PEP 3333. All of the functions taking an *environ* parameter expect a WSGI-compliant dictionary to be supplied; please see PEP 3333 for a detailed specification.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 49)

Unknown directive type "function".

```
.. function:: guess_scheme(environ)

   Return a guess for whether ``wsgi.url_scheme`` should be "http" or "https", by
   checking for a ``HTTPS`` environment variable in the *environ* dictionary.  The
   return value is a string.

   This function is useful when creating a gateway that wraps CGI or a CGI-like
   protocol such as FastCGI.  Typically, servers providing such protocols will
   include a ``HTTPS`` variable with a value of "1", "yes", or "on" when a request
   is received via SSL.  So, this function returns "https" if such a value is
   found, and "http" otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 62)

Unknown directive type "function".

```
.. function:: request_uri(environ, include_query=True)

   Return the full request URI, optionally including the query string, using the
   algorithm found in the "URL Reconstruction" section of :pep:`3333`.  If
   *include_query* is false, the query string is not included in the resulting URI.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 69)

Unknown directive type "function".

```
.. function:: application_uri(environ)

   Similar to :func:`request_uri`, except that the ``PATH_INFO`` and
   ``QUERY_STRING`` variables are ignored.  The result is the base URI of the
   application object addressed by the request.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 76)

Unknown directive type "function".

```
.. function:: shift_path_info(environ)

   Shift a single name from ``PATH_INFO`` to ``SCRIPT_NAME`` and return the name.
   The *environ* dictionary is *modified* in-place; use a copy if you need to keep
   the original ``PATH_INFO`` or ``SCRIPT_NAME`` intact.

   If there are no remaining path segments in ``PATH_INFO``, ``None`` is returned.

   Typically, this routine is used to process each portion of a request URI path,
   for example to treat the path as a series of dictionary keys. This routine
   modifies the passed-in environment to make it suitable for invoking another WSGI
   application that is located at the target URI. For example, if there is a WSGI
   application at ``/foo``, and the request URI path is ``/foo/bar/baz``, and the
   WSGI application at ``/foo`` calls :func:`shift_path_info`, it will receive the
   string "bar", and the environment will be updated to be suitable for passing to
   a WSGI application at ``/foo/bar``.  That is, ``SCRIPT_NAME`` will change from
   ``/foo`` to ``/foo/bar``, and ``PATH_INFO`` will change from ``/bar/baz`` to
   ``/baz``.

   When ``PATH_INFO`` is just a "/", this routine returns an empty string and
   appends a trailing slash to ``SCRIPT_NAME``, even though empty path segments are
   normally ignored, and ``SCRIPT_NAME`` doesn't normally end in a slash.  This is
   intentional behavior, to ensure that an application can tell the difference
   between URIs ending in ``/x`` from ones ending in ``/x/`` when using this
   routine to do object traversal.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 103)

Unknown directive type "function".

```
.. function:: setup_testing_defaults(environ)

   Update *environ* with trivial defaults for testing purposes.

   This routine adds various parameters required for WSGI, including ``HTTP_HOST``,
   ``SERVER_NAME``, ``SERVER_PORT``, ``REQUEST_METHOD``, ``SCRIPT_NAME``,
   ``PATH_INFO``, and all of the :pep:`3333`\ -defined ``wsgi.*`` variables.  It
   only supplies default values, and does not replace any existing settings for
   these variables.

   This routine is intended to make it easier for unit tests of WSGI servers and
   applications to set up dummy environments.  It should NOT be used by actual WSGI
   servers or applications, since the data is fake!

   Example usage::

      from wsgiref.util import setup_testing_defaults
      from wsgiref.simple_server import make_server

      # A relatively simple WSGI application. It's going to print out the
      # environment dictionary after being updated by setup_testing_defaults
      def simple_app(environ, start_response):
          setup_testing_defaults(environ)

          status = '200 OK'
          headers = [('Content-type', 'text/plain; charset=utf-8')]

          start_response(status, headers)

          ret = [("%s: %s\n" % (key, value)).encode("utf-8")
                 for key, value in environ.items()]
          return ret

      with make_server('', 8000, simple_app) as httpd:
          print("Serving on port 8000...")
          httpd.serve_forever()
```

In addition to the environment functions above, the :mod:`wsgiref.util` module also provides these miscellaneous utilities:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 141); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 145)

Unknown directive type "function".

```
.. function:: is_hop_by_hop(header_name)

   Return ``True`` if 'header_name' is an HTTP/1.1 "Hop-by-Hop" header, as defined by
   :rfc:`2616`.
```

A wrapper to convert a file-like object to an :term:`iterator`. The resulting objects are :term:`iterable`s. As the object is iterated over, the optional *blksize* parameter will be repeatedly passed to the *filelike* object's :meth:`read` method to obtain bytestrings to yield. When :meth:`read` returns an empty bytestring, iteration is ended and is not resumable.

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

Unknown interpreted text role "term".

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

Unknown interpreted text role "term".

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

Unknown interpreted text role "meth".

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

Unknown interpreted text role "meth".

If *filelike* has a :meth:`close` method, the returned object will also have a :meth:`close` method, and it will invoke the *filelike* object's :meth:`close` method when called.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 159); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 159); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 159); *backlink*

Unknown interpreted text role "meth".

Example usage:

```
from io import StringIO
from wsgiref.util import FileWrapper

# We're using a StringIO-buffer for as the file-like object
filelike = StringIO("This is an example file-like object"*10)
wrapper = FileWrapper(filelike, blksize=5)

for chunk in wrapper:
    print(chunk)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 175)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.11
   Support for :meth:`__getitem__` method has been removed.
```

:mod:`wsgiref.headers` -- WSGI response header tools

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

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 182)

Unknown directive type "module".

```
.. module:: wsgiref.headers
   :synopsis: WSGI response header tools.
```

This module provides a single class, :class:`Headers`, for convenient manipulation of WSGI response headers using a mapping-like interface.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 186); *backlink*

Unknown interpreted text role "class".

Create a mapping-like object wrapping *headers*, which must be a list of header name/value tuples as described in PEP 3333. The default value of *headers* is an empty list.

:class:`Headers` objects support typical mapping operations including :meth:`__getitem__`, :meth:`get`, :meth:`__setitem__`, :meth:`setdefault`, :meth:`__delitem__` and :meth:`__contains__`. For each of these methods, the key is the header name (treated case-insensitively), and the value is the first value associated with that header name. Setting a header deletes any existing values for that header, then adds a new value at the end of the wrapped header list. Headers' existing order is generally maintained, with new headers added to the end of the wrapped list.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 196**); ***backlink***

Unknown interpreted text role "meth".

Unlike a dictionary, :class:`Headers` objects do not raise an error when you try to get or delete a key that isn't in the wrapped header list. Getting a nonexistent header just returns `None`, and deleting a nonexistent header does nothing.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 205**); ***backlink***

Unknown interpreted text role "class".

:class:`Headers` objects also support :meth:`keys`, :meth:`values`, and :meth:`items` methods. The lists returned by :meth:`keys` and :meth:`items` can include the same key more than once if there is a multi-valued header. The `len()` of a :class:`Headers` object is the same as the length of its :meth:`items`, which is the same as the length of the wrapped header list. In fact, the :meth:`items` method just returns a copy of the wrapped header list.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 210**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 210**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 210); *backlink*

Unknown interpreted text role "meth".

Calling `bytes()` on a :class:`Headers` object returns a formatted bytestring suitable for transmission as HTTP response headers. Each header is placed on a line with its value, separated by a colon and a space. Each line is terminated by a carriage return and line feed, and the bytestring is terminated with a blank line.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 217); *backlink*

Unknown interpreted text role "class".

In addition to their mapping interface and formatting features, :class:`Headers` objects also have the following methods for querying and adding multi-valued headers, and for adding headers with MIME parameters:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 223); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 228)

Unknown directive type "method".

```
.. method:: Headers.get_all(name)

   Return a list of all the values for the named header.

   The returned list will be sorted in the order they appeared in the original
   header list or were added to this instance, and may contain duplicates.  Any
   fields deleted and re-inserted are always appended to the header list.  If no
   fields exist with the given name, returns an empty list.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 238)

Unknown directive type "method".

```
.. method:: Headers.add_header(name, value, **_params)

   Add a (possibly multi-valued) header, with optional MIME parameters specified
   via keyword arguments.

   *name* is the header field to add.  Keyword arguments can be used to set MIME
   parameters for the header field.  Each parameter must be a string or ``None``.
   Underscores in parameter names are converted to dashes, since dashes are illegal
   in Python identifiers, but many MIME parameter names include dashes.  If the
   parameter value is a string, it is added to the header value parameters in the
   form ``name="value"``. If it is ``None``, only the parameter name is added.
   (This is used for MIME parameters without a value.)  Example usage::

      h.add_header('content-disposition', 'attachment', filename='bud.gif')

   The above will add a header that looks like this::

      Content-Disposition: attachment; filename="bud.gif"
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 258)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.5
   *headers* parameter is optional.
```

:mod:`wsgiref.simple_server` -- a simple WSGI HTTP server

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

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 265)

Unknown directive type "module".

```
.. module:: wsgiref.simple_server
   :synopsis: A simple WSGI HTTP server.
```

This module implements a simple HTTP server (based on :mod:`http.server`) that serves WSGI applications. Each server instance serves a single WSGI application on a given host and port. If you want to serve multiple applications on a single host and port, you should create a WSGI application that parses `PATH_INFO` to select which application to invoke for each request. (E.g., using the :func:`shift_path_info` function from :mod:`wsgiref.util`.)

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 269); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 269); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 269); *backlink*

Unknown interpreted text role "mod".

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

main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 278)

Unknown directive type "function".

```
.. function:: make_server(host, port, app, server_class=WSGIServer, handler_class=WSGIRequestHandle

   Create a new WSGI server listening on *host* and *port*, accepting connections
   for *app*.  The return value is an instance of the supplied *server_class*, and
   will process requests using the specified *handler_class*.  *app* must be a WSGI
   application object, as defined by :pep:`3333`.

   Example usage::

      from wsgiref.simple_server import make_server, demo_app

      with make_server('', 8000, demo_app) as httpd:
          print("Serving HTTP on port 8000...")

          # Respond to requests until process is killed
          httpd.serve_forever()

          # Alternative: serve one request, then exit
          httpd.handle_request()
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 299)

Unknown directive type "function".

```
.. function:: demo_app(environ, start_response)

   This function is a small but complete WSGI application that returns a text page
   containing the message "Hello world!" and a list of the key/value pairs provided
   in the *environ* parameter.  It's useful for verifying that a WSGI server (such
   as :mod:`wsgiref.simple_server`) is able to run a simple WSGI application
   correctly.
```

Create a :class:`WSGIServer` instance. *server_address* should be a `(host,port)` tuple, and *RequestHandlerClass* should be the subclass of :class:`http.server.BaseHTTPRequestHandler` that will be used to process requests.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 310); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 310); *backlink*

Unknown interpreted text role "class".

You do not normally need to call this constructor, as the :func:`make_server` function can handle all the details for you.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 315); *backlink*

Unknown interpreted text role "func".

:class:`WSGIServer` is a subclass of :class:`http.server.HTTPServer`, so all of its methods (such as :meth:`serve_forever` and :meth:`handle_request`) are available. :class:`WSGIServer` also provides these WSGI-specific methods:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 318); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 318); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 318); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 318); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 318); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 323)

Unknown directive type "method".

```
.. method:: WSGIServer.set_app(application)

   Sets the callable *application* as the WSGI application that will receive
   requests.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 329)

Unknown directive type "method".

```
.. method:: WSGIServer.get_app()

   Returns the currently-set application callable.
```

Normally, however, you do not need to use these additional methods, as :meth:`set_app` is normally called by :func:`make_server`, and the :meth:`get_app` exists mainly for the benefit of request handler instances.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 333); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 333); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 333); *backlink*

Unknown interpreted text role "meth".

Create an HTTP handler for the given *request* (i.e. a socket), *client_address* (a `(host,port)` tuple), and *server* (:class:`WSGIServer` instance).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 340); *backlink*

Unknown interpreted text role "class".

You do not need to create instances of this class directly; they are automatically created as needed by :class:`WSGIServer` objects. You can, however, subclass this class and supply it as a *handler_class* to the :func:`make_server` function. Some possibly relevant methods for overriding in subclasses:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 343); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 343); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 350)

Unknown directive type "method".

```
.. method:: WSGIRequestHandler.get_environ()

   Returns a dictionary containing the WSGI environment for a request.  The default
   implementation copies the contents of the :class:`WSGIServer` object's
   :attr:`base_environ` dictionary attribute and then adds various headers derived
   from the HTTP request.  Each call to this method should return a new dictionary
   containing all of the relevant CGI environment variables as specified in
   :pep:`3333`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 360)

Unknown directive type "method".

```
.. method:: WSGIRequestHandler.get_stderr()

   Return the object that should be used as the ``wsgi.errors`` stream. The default
   implementation just returns ``sys.stderr``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 366)

Unknown directive type "method".

```
.. method:: WSGIRequestHandler.handle()

   Process the HTTP request.  The default implementation creates a handler instance
   using a :mod:`wsgiref.handlers` class to implement the actual WSGI application
   interface.
```

:mod:`wsgiref.validate` --- WSGI conformance checker

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 373); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 376)

Unknown directive type "module".

```
.. module:: wsgiref.validate
   :synopsis: WSGI conformance checker.
```

When creating new WSGI application objects, frameworks, servers, or middleware, it can be useful to validate the new code's conformance using :mod:`wsgiref.validate`. This module provides a function that creates WSGI application objects that validate communications between a WSGI server or gateway and a WSGI application object, to check both sides for protocol conformance.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 380); *backlink*

Unknown interpreted text role "mod".

Note that this utility does not guarantee complete PEP 3333 compliance; an absence of errors from this module does not necessarily mean that errors do not exist. However, if this module does produce an error, then it is virtually certain that either the server or application is not 100% compliant.

This module is based on the :mod:`paste.lint` module from Ian Bicking's "Python Paste" library.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 392); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 396)

Unknown directive type "function".

```
.. function:: validator(application)

   Wrap *application* and return a new WSGI application object.  The returned
   application will forward all requests to the original *application*, and will
   check that both the *application* and the server invoking it are conforming to
   the WSGI specification and to :rfc:`2616`.

   Any detected nonconformance results in an :exc:`AssertionError` being raised;
   note, however, that how these errors are handled is server-dependent.  For
   example, :mod:`wsgiref.simple_server` and other servers based on
   :mod:`wsgiref.handlers` (that don't override the error handling methods to do
   something else) will simply output a message that an error has occurred, and
   dump the traceback to ``sys.stderr`` or some other error stream.

   This wrapper may also generate output using the :mod:`warnings` module to
   indicate behaviors that are questionable but which may not actually be
   prohibited by :pep:`3333`.  Unless they are suppressed using Python command-line
   options or the :mod:`warnings` API, any such warnings will be written to
   ``sys.stderr`` (*not* ``wsgi.errors``, unless they happen to be the same
   object).

   Example usage::

      from wsgiref.validate import validator
      from wsgiref.simple_server import make_server

      # Our callable object which is intentionally not compliant to the
      # standard, so the validator is going to break
      def simple_app(environ, start_response):
          status = '200 OK'  # HTTP Status
          headers = [('Content-type', 'text/plain')]  # HTTP Headers
          start_response(status, headers)

          # This is going to break because we need to return a list, and
          # the validator is going to inform us
          return b"Hello World"

      # This is the application wrapped in a validator
      validator_app = validator(simple_app)

      with make_server('', 8000, validator_app) as httpd:
          print("Listening on port 8000....")
          httpd.serve_forever()
```

:mod:`wsgiref.handlers` -- server/gateway base classes

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

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 444)

Unknown directive type "module".

```
.. module:: wsgiref.handlers
   :synopsis: WSGI server/gateway base classes.
```

This module provides base handler classes for implementing WSGI servers and gateways. These base classes handle most of the work of communicating with a WSGI application, as long as they are given a CGI-like environment, along with input, output, and error streams.

CGI-based invocation via `sys.stdin`, `sys.stdout`, `sys.stderr` and `os.environ`. This is useful when you have a WSGI application and want to run it as a CGI script. Simply invoke `CGIHandler().run(app)`, where `app` is the WSGI application object you wish to invoke.

This class is a subclass of :class:`BaseCGIHandler` that sets `wsgi.run_once` to true, `wsgi.multithread` to false, and `wsgi.multiprocess` to true, and always uses :mod:`sys` and :mod:`os` to obtain the necessary CGI streams and environment.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 461); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 461); ***backlink***

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 461); ***backlink***

Unknown interpreted text role "mod".

A specialized alternative to :class:`CGIHandler`, for use when deploying on Microsoft's IIS web server, without having set the config allowPathInfo option (IIS>=7) or metabase allowPathInfoForScriptMappings (IIS<7).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 469); ***backlink***

Unknown interpreted text role "class".

By default, IIS gives a `PATH_INFO` that duplicates the `SCRIPT_NAME` at the front, causing problems for WSGI applications that wish to implement routing. This handler strips any such duplicated path.

IIS can be configured to pass the correct `PATH_INFO`, but this causes another bug where `PATH_TRANSLATED` is wrong. Luckily this variable is rarely used and is not guaranteed by WSGI. On IIS<7, though, the setting can only be made on a vhost level, affecting all other script mappings, many of which break when exposed to the `PATH_TRANSLATED` bug. For this reason IIS<7 is almost never deployed with the fix (Even IIS7 rarely uses it because there is still no UI for it.).

There is no way for CGI code to tell whether the option was set, so a separate handler class is provided. It is used in the same way as :class:`CGIHandler`, i.e., by calling `IISCGIHandler().run(app)`, where `app` is the WSGI application object you wish to invoke.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 485); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 490)

Unknown directive type "versionadded".

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

Similar to :class:`CGIHandler`, but instead of using the :mod:`sys` and :mod:`os` modules, the CGI environment and I/O streams are specified explicitly. The *multithread* and *multiprocess* values are used to set the `wsgi.multithread` and `wsgi.multiprocess` flags for any applications run by the handler instance.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 495); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 495); ***backlink***

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 495**); ***backlink***

Unknown interpreted text role "mod".

This class is a subclass of :class:`SimpleHandler` intended for use with software other than HTTP "origin servers". If you are writing a gateway protocol implementation (such as CGI, FastCGI, SCGI, etc.) that uses a `Status:` header to send an HTTP status, you probably want to subclass this instead of :class:`SimpleHandler`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 501**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 501**); ***backlink***

Unknown interpreted text role "class".

Similar to :class:`BaseCGIHandler`, but designed for use with HTTP origin servers. If you are writing an HTTP server implementation, you will probably want to subclass this instead of :class:`BaseCGIHandler`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 510**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 510**); ***backlink***

Unknown interpreted text role "class".

This class is a subclass of :class:`BaseHandler`. It overrides the :meth:`__init__`, :meth:`get_stdin`, :meth:`get_stderr`, :meth:`add_cgi_vars`, :meth:`_write`, and :meth:`_flush` methods to support explicitly setting the environment and streams via the constructor. The supplied environment and streams are stored in the :attr:`stdin`, :attr:`stdout`, :attr:`stderr`, and :attr:`environ` attributes.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 514**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 514**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 514**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 514**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 514**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, **line 514**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 514); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 514); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 514); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 514); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 514); *backlink*

Unknown interpreted text role "attr".

The :meth:`~io.BufferedIOBase.write` method of *stdout* should write each chunk in full, like :class:`io.BufferedIOBase`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 521); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 521); *backlink*

Unknown interpreted text role "class".

This is an abstract base class for running WSGI applications. Each instance will handle a single HTTP request, although in principle you could create a subclass that was reusable for multiple requests.

:class:`BaseHandler` instances have only one method intended for external use:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 531); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 534)

Unknown directive type "method".

```
.. method:: BaseHandler.run(app)

   Run the specified WSGI application, *app*.
```

All of the other :class:`BaseHandler` methods are invoked by this method in the process of running the application, and thus exist primarily to allow customizing the process.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 538); *backlink*

Unknown interpreted text role "class".

The following methods MUST be overridden in a subclass:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 545)

Unknown directive type "method".

```
.. method:: BaseHandler._write(data)

   Buffer the bytes *data* for transmission to the client.  It's okay if this
   method actually transmits the data; :class:`BaseHandler` just separates write
   and flush operations for greater efficiency when the underlying system actually
   has such a distinction.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 553)

Unknown directive type "method".

```
.. method:: BaseHandler._flush()

   Force buffered data to be transmitted to the client.  It's okay if this method
   is a no-op (i.e., if :meth:`_write` actually sends the data).
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 559)

Unknown directive type "method".

```
.. method:: BaseHandler.get_stdin()

   Return an input stream object suitable for use as the ``wsgi.input`` of the
   request currently being processed.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 565)

Unknown directive type "method".

```
.. method:: BaseHandler.get_stderr()

   Return an output stream object suitable for use as the ``wsgi.errors`` of the
   request currently being processed.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 571)

Unknown directive type "method".

```
.. method:: BaseHandler.add_cgi_vars()

   Insert CGI variables for the current request into the :attr:`environ` attribute.
```

Here are some other methods and attributes you may wish to override. This list is only a summary, however, and does not include every method that can be overridden. You should consult the docstrings and source code for additional information before attempting to create a customized :class:`BaseHandler` subclass.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 575); *backlink*

Unknown interpreted text role "class".

Attributes and methods for customizing the WSGI environment:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 584)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.wsgi_multithread
```

```
   The value to be used for the ``wsgi.multithread`` environment variable.  It
   defaults to true in :class:`BaseHandler`, but may have a different default (or
   be set by the constructor) in the other subclasses.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 591)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.wsgi_multiprocess

   The value to be used for the ``wsgi.multiprocess`` environment variable.  It
   defaults to true in :class:`BaseHandler`, but may have a different default (or
   be set by the constructor) in the other subclasses.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 598)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.wsgi_run_once

   The value to be used for the ``wsgi.run_once`` environment variable.  It
   defaults to false in :class:`BaseHandler`, but :class:`CGIHandler` sets it to
   true by default.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 605)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.os_environ

   The default environment variables to be included in every request's WSGI
   environment.  By default, this is a copy of ``os.environ`` at the time that
   :mod:`wsgiref.handlers` was imported, but subclasses can either create their own
   at the class or instance level.  Note that the dictionary should be considered
   read-only, since the default value is shared between multiple classes and
   instances.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 615)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.server_software

   If the :attr:`origin_server` attribute is set, this attribute's value is used to
   set the default ``SERVER_SOFTWARE`` WSGI environment variable, and also to set a
   default ``Server:`` header in HTTP responses.  It is ignored for handlers (such
   as :class:`BaseCGIHandler` and :class:`CGIHandler`) that are not HTTP origin
   servers.

   .. versionchanged:: 3.3
      The term "Python" is replaced with implementation specific term like
      "CPython", "Jython" etc.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 627)

Unknown directive type "method".

```
.. method:: BaseHandler.get_scheme()

   Return the URL scheme being used for the current request.  The default
   implementation uses the :func:`guess_scheme` function from :mod:`wsgiref.util`
   to guess whether the scheme should be "http" or "https", based on the current
   request's :attr:`environ` variables.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 635)

Unknown directive type "method".

```
.. method:: BaseHandler.setup_environ()

   Set the :attr:`environ` attribute to a fully-populated WSGI environment.  The
   default implementation uses all of the above methods and attributes, plus the
   :meth:`get_stdin`, :meth:`get_stderr`, and :meth:`add_cgi_vars` methods and the
   :attr:`wsgi_file_wrapper` attribute.  It also inserts a ``SERVER_SOFTWARE`` key
   if not present, as long as the :attr:`origin_server` attribute is a true value
   and the :attr:`server_software` attribute is set.
```

Methods and attributes for customizing exception handling:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 647)

Unknown directive type "method".

```
.. method:: BaseHandler.log_exception(exc_info)

   Log the *exc_info* tuple in the server log.  *exc_info* is a ``(type, value,
   traceback)`` tuple.  The default implementation simply writes the traceback to
   the request's ``wsgi.errors`` stream and flushes it.  Subclasses can override
   this method to change the format or retarget the output, mail the traceback to
   an administrator, or whatever other action may be deemed suitable.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 656)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.traceback_limit

   The maximum number of frames to include in tracebacks output by the default
   :meth:`log_exception` method.  If ``None``, all frames are included.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 662)

Unknown directive type "method".

```
.. method:: BaseHandler.error_output(environ, start_response)

   This method is a WSGI application to generate an error page for the user.  It is
   only invoked if an error occurs before headers are sent to the client.

   This method can access the current error information using ``sys.exc_info()``,
   and should pass that information to *start_response* when calling it (as
   described in the "Error Handling" section of :pep:`3333`).

   The default implementation just uses the :attr:`error_status`,
   :attr:`error_headers`, and :attr:`error_body` attributes to generate an output
   page.  Subclasses can override this to produce more dynamic error output.

   Note, however, that it's not recommended from a security perspective to spit out
   diagnostics to any old user; ideally, you should have to do something special to
   enable diagnostic output, which is why the default implementation doesn't
   include any.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst, line 681)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.error_status

   The HTTP status used for error responses.  This should be a status string as
   defined in :pep:`3333`; it defaults to a 500 code and message.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 687)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.error_headers

   The HTTP headers used for error responses.  This should be a list of WSGI
   response headers (``(name, value)`` tuples), as described in :pep:`3333`.  The
   default list just sets the content type to ``text/plain``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 694)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.error_body

   The error response body.  This should be an HTTP response body bytestring. It
   defaults to the plain text, "A server error occurred.  Please contact the
   administrator."
```

Methods and attributes for PEP 3333's "Optional Platform-Specific File Handling" feature:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 704)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.wsgi_file_wrapper

   A ``wsgi.file_wrapper`` factory, or ``None``.  The default value of this
   attribute is the :class:`wsgiref.util.FileWrapper` class.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 710)

Unknown directive type "method".

```
.. method:: BaseHandler.sendfile()

   Override to implement platform-specific file transmission.  This method is
   called only if the application's return value is an instance of the class
   specified by the :attr:`wsgi_file_wrapper` attribute.  It should return a true
   value if it was able to successfully transmit the file, so that the default
   transmission code will not be executed. The default implementation of this
   method just returns a false value.
```

Miscellaneous methods and attributes:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 722)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.origin_server

   This attribute should be set to a true value if the handler's :meth:`_write` and
   :meth:`_flush` are being used to communicate directly to the client, rather than
   via a CGI-like gateway protocol that wants the HTTP status in a special
   ``Status:`` header.

   This attribute's default value is true in :class:`BaseHandler`, but false in
   :class:`BaseCGIHandler` and :class:`CGIHandler`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 733)

Unknown directive type "attribute".

```
.. attribute:: BaseHandler.http_version

   If :attr:`origin_server` is true, this string attribute is used to set the HTTP
   version of the response set to the client.  It defaults to ``"1.0"``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 739)

Unknown directive type "function".

```
.. function:: read_environ()

   Transcode CGI variables from ``os.environ`` to :pep:`3333` "bytes in unicode"
   strings, returning a new dictionary.  This function is used by
   :class:`CGIHandler` and :class:`IISCGIHandler` in place of directly using
   ``os.environ``, which is not necessarily WSGI-compliant on all platforms
   and web servers using Python 3 -- specifically, ones where the OS's
   actual environment is Unicode (i.e. Windows), or ones where the environment
   is bytes, but the system encoding used by Python to decode it is anything
   other than ISO-8859-1 (e.g. Unix systems using UTF-8).

   If you are implementing a CGI-based handler of your own, you probably want
   to use this routine instead of just copying values out of ``os.environ``
   directly.

   .. versionadded:: 3.2
```

Examples

This is a working "Hello World" WSGI application:

```
from wsgiref.simple_server import make_server

# Every WSGI application must have an application object - a callable
# object that accepts two arguments. For that purpose, we're going to
# use a function (note that you're not limited to a function, you can
# use a class for example). The first argument passed to the function
# is a dictionary containing CGI-style environment variables and the
# second variable is the callable object.
def hello_world_app(environ, start_response):
    status = '200 OK'  # HTTP Status
    headers = [('Content-type', 'text/plain; charset=utf-8')]  # HTTP Headers
    start_response(status, headers)

    # The returned object is going to be printed
    return [b"Hello World"]

with make_server('', 8000, hello_world_app) as httpd:
    print("Serving on port 8000...")

    # Serve until process is killed
    httpd.serve_forever()
```

Example of a WSGI application serving the current directory, accept optional directory and port number (default: 8000) on the command line:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]wsgiref.rst`, line 788)

Unknown directive type "literalinclude".

```
.. literalinclude:: ../../Tools/scripts/serve.py
```