:mod:`queue` --- A synchronized queue class

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

Unknown directive type "module".

```
.. module:: queue
   :synopsis: A synchronized queue class.
```

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

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

Unknown interpreted text role "source".

The :mod:`queue` module implements multi-producer, multi-consumer queues. It is especially useful in threaded programming when information must be exchanged safely between multiple threads. The :class:`Queue` class in this module implements all the required locking semantics.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 11); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 11); *backlink*

Unknown interpreted text role "class".

The module implements three types of queue, which differ only in the order in which the entries are retrieved. In a :abbr:`FIFO (first-in, first-out)` queue, the first tasks added are the first retrieved. In a :abbr:`LIFO (last-in, first-out)` queue, the most recently added entry is the first retrieved (operating like a stack). With a priority queue, the entries are kept sorted (using the :mod:`heapq` module) and the lowest valued entry is retrieved first.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 16); *backlink*

Unknown interpreted text role "abbr".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 16); *backlink*

Unknown interpreted text role "abbr".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 16); *backlink*

Unknown interpreted text role "mod".

Internally, those three types of queues use locks to temporarily block competing threads; however, they are not designed to handle reentrancy within a thread.

In addition, the module implements a "simple" :abbr:`FIFO (first-in, first-out)` queue type, :class:`SimpleQueue`, whose specific implementation provides additional guarantees in exchange for the smaller functionality.

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

Unknown interpreted text role "abbr".

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

Unknown interpreted text role "class".

The :mod:`queue` module defines the following classes and exceptions:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 33); *backlink*

Unknown interpreted text role "mod".

Constructor for a :abbr:`FIFO (first-in, first-out)` queue. *maxsize* is an integer that sets the upperbound limit on the number of items that can be placed in the queue. Insertion will block once this size has been reached, until queue items are consumed. If *maxsize* is less than or equal to zero, the queue size is infinite.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 37); *backlink*

Unknown interpreted text role "abbr".

Constructor for a :abbr:`LIFO (last-in, first-out)` queue. *maxsize* is an integer that sets the upperbound limit on the number of items that can be placed in the queue. Insertion will block once this size has been reached, until queue items are consumed. If *maxsize* is less than or equal to zero, the queue size is infinite.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 45); *backlink*

Unknown interpreted text role "abbr".

Constructor for a priority queue. *maxsize* is an integer that sets the upperbound limit on the number of items that can be placed in the queue. Insertion will block once this size has been reached, until queue items are consumed. If *maxsize* is less than or equal to zero, the queue size is infinite.

The lowest valued entries are retrieved first (the lowest valued entry is the one returned by `sorted(list(entries))[0]`). A typical pattern for entries is a tuple in the form: `(priority_number, data)`.

If the *data* elements are not comparable, the data can be wrapped in a class that ignores the data item and only compares the priority number:

```
from dataclasses import dataclass, field
from typing import Any

@dataclass(order=True)
class PrioritizedItem:
    priority: int
    item: Any=field(compare=False)
```

Constructor for an unbounded :abbr:`FIFO (first-in, first-out)` queue. Simple queues lack advanced functionality such as task tracking.

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

Unknown interpreted text role "abbr".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 79)

Unknown directive type "versionadded".

```
.. versionadded:: 3.7
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 82)

Unknown directive type "exception".

```
.. exception:: Empty

   Exception raised when non-blocking :meth:`~Queue.get` (or
   :meth:`~Queue.get_nowait`) is called
   on a :class:`Queue` object which is empty.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 89)

Unknown directive type "exception".

```
.. exception:: Full

   Exception raised when non-blocking :meth:`~Queue.put` (or
   :meth:`~Queue.put_nowait`) is called
   on a :class:`Queue` object which is full.
```

Queue Objects

Queue objects (:class:`Queue`, :class:`LifoQueue`, or :class:`PriorityQueue`) provide the public methods described below.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 101); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 101); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 101); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 105)

Unknown directive type "method".

```
.. method:: Queue.qsize()

   Return the approximate size of the queue.  Note, qsize() > 0 doesn't
   guarantee that a subsequent get() will not block, nor will qsize() < maxsize
   guarantee that put() will not block.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 112)

Unknown directive type "method".

```
.. method:: Queue.empty()

   Return ``True`` if the queue is empty, ``False`` otherwise.  If empty()
   returns ``True`` it doesn't guarantee that a subsequent call to put()
   will not block.  Similarly, if empty() returns ``False`` it doesn't
   guarantee that a subsequent call to get() will not block.
```

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

Unknown directive type "method".

```
.. method:: Queue.full()

   Return ``True`` if the queue is full, ``False`` otherwise.  If full()
   returns ``True`` it doesn't guarantee that a subsequent call to get()
   will not block.  Similarly, if full() returns ``False`` it doesn't
   guarantee that a subsequent call to put() will not block.
```

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

Unknown directive type "method".

```
.. method:: Queue.put(item, block=True, timeout=None)

   Put *item* into the queue. If optional args *block* is true and *timeout* is
   ``None`` (the default), block if necessary until a free slot is available. If
   *timeout* is a positive number, it blocks at most *timeout* seconds and raises
   the :exc:`Full` exception if no free slot was available within that time.
   Otherwise (*block* is false), put an item on the queue if a free slot is
   immediately available, else raise the :exc:`Full` exception (*timeout* is
   ignored in that case).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 139)

Unknown directive type "method".

```
.. method:: Queue.put_nowait(item)

   Equivalent to ``put(item, False)``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 144)

Unknown directive type "method".

```
.. method:: Queue.get(block=True, timeout=None)

   Remove and return an item from the queue. If optional args *block* is true and
   *timeout* is ``None`` (the default), block if necessary until an item is available.
   If *timeout* is a positive number, it blocks at most *timeout* seconds and
   raises the :exc:`Empty` exception if no item was available within that time.
   Otherwise (*block* is false), return an item if one is immediately available,
   else raise the :exc:`Empty` exception (*timeout* is ignored in that case).

   Prior to 3.0 on POSIX systems, and for all versions on Windows, if
   *block* is true and *timeout* is ``None``, this operation goes into
   an uninterruptible wait on an underlying lock. This means that no exceptions
   can occur, and in particular a SIGINT will not trigger a :exc:`KeyboardInterrupt`.
```

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

Unknown directive type "method".

```
.. method:: Queue.get_nowait()

   Equivalent to ``get(False)``.
```

Two methods are offered to support tracking whether enqueued tasks have been fully processed by daemon consumer threads.

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

main\Doc\library\[cpython-main][Doc][library]queue.rst, line 167)

Unknown directive type "method".

```
.. method:: Queue.task_done()

   Indicate that a formerly enqueued task is complete.  Used by queue consumer
   threads.  For each :meth:`get` used to fetch a task, a subsequent call to
   :meth:`task_done` tells the queue that the processing on the task is complete.

   If a :meth:`join` is currently blocking, it will resume when all items have been
   processed (meaning that a :meth:`task_done` call was received for every item
   that had been :meth:`put` into the queue).

   Raises a :exc:`ValueError` if called more times than there were items placed in
   the queue.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 181)

Unknown directive type "method".

```
.. method:: Queue.join()

   Blocks until all items in the queue have been gotten and processed.

   The count of unfinished tasks goes up whenever an item is added to the queue.
   The count goes down whenever a consumer thread calls :meth:`task_done` to
   indicate that the item was retrieved and all work on it is complete. When the
   count of unfinished tasks drops to zero, :meth:`join` unblocks.
```

Example of how to wait for enqueued tasks to be completed:

```
import threading, queue

q = queue.Queue()

def worker():
    while True:
        item = q.get()
        print(f'Working on {item}')
        print(f'Finished {item}')
        q.task_done()

# Turn-on the worker thread.
threading.Thread(target=worker, daemon=True).start()

# Send thirty task requests to the worker.
for item in range(30):
    q.put(item)

# Block until all tasks are done.
q.join()
print('All work completed')
```

SimpleQueue Objects

:class:`SimpleQueue` objects provide the public methods described below.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 219); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 221)

Unknown directive type "method".

```
.. method:: SimpleQueue.qsize()

   Return the approximate size of the queue.  Note, qsize() > 0 doesn't
   guarantee that a subsequent get() will not block.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 227)

Unknown directive type "method".

```
.. method:: SimpleQueue.empty()

   Return ``True`` if the queue is empty, ``False`` otherwise. If empty()
   returns ``False`` it doesn't guarantee that a subsequent call to get()
   will not block.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 234)

Unknown directive type "method".

```
.. method:: SimpleQueue.put(item, block=True, timeout=None)

   Put *item* into the queue.  The method never blocks and always succeeds
   (except for potential low-level errors such as failure to allocate memory).
   The optional args *block* and *timeout* are ignored and only provided
   for compatibility with :meth:`Queue.put`.

   .. impl-detail::
      This method has a C implementation which is reentrant.  That is, a
      ``put()`` or ``get()`` call can be interrupted by another ``put()``
      call in the same thread without deadlocking or corrupting internal
      state inside the queue.  This makes it appropriate for use in
      destructors such as ``__del__`` methods or :mod:`weakref` callbacks.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 249)

Unknown directive type "method".

```
.. method:: SimpleQueue.put_nowait(item)

   Equivalent to ``put(item)``, provided for compatibility with
   :meth:`Queue.put_nowait`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 255)

Unknown directive type "method".

```
.. method:: SimpleQueue.get(block=True, timeout=None)

   Remove and return an item from the queue.  If optional args *block* is true and
   *timeout* is ``None`` (the default), block if necessary until an item is available.
   If *timeout* is a positive number, it blocks at most *timeout* seconds and
   raises the :exc:`Empty` exception if no item was available within that time.
   Otherwise (*block* is false), return an item if one is immediately available,
   else raise the :exc:`Empty` exception (*timeout* is ignored in that case).
```

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

Unknown directive type "method".

```
.. method:: SimpleQueue.get_nowait()

   Equivalent to ``get(False)``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]queue.rst, line 270)

Unknown directive type "seealso".

```
.. seealso::

   Class :class:`multiprocessing.Queue`
      A queue class for use in a multi-processing (rather than multi-threading)
      context.

   :class:`collections.deque` is an alternative implementation of unbounded
   queues with fast atomic :meth:`~collections.deque.append` and
   :meth:`~collections.deque.popleft` operations that do not require locking
   and also support indexing.
```