Module: tf.queue

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue#top_of_page)
* [Classes](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue#classes)

Public API for tf.queue namespace.

Classes

[class FIFOQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue): A queue implementation that dequeues elements in first-in first-out order.

[class PaddingFIFOQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PaddingFIFOQueue): A FIFOQueue that supports batching variable-sized tensors by padding.

[class PriorityQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PriorityQueue): A queue implementation that dequeues elements in prioritized order.

[class QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase): Base class for queue implementations.

[class RandomShuffleQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue): A queue implementation that dequeues elements in a random order.

# tf.queue.FIFOQueue

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue#top_of_page)
* [Class FIFOQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue#class_fifoqueue)
  + [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue#aliases)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue#properties)

## Class FIFOQueue

A queue implementation that dequeues elements in first-in first-out order.

Inherits From: [QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase)

### Aliases:

* Class tf.compat.v1.FIFOQueue
* Class tf.compat.v1.queue.FIFOQueue
* Class tf.compat.v2.queue.FIFOQueue
* Class tf.queue.FIFOQueue

Defined in [python/ops/data\_flow\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/data_flow_ops.py).

See [tf.queue.QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase) for a description of the methods on this class.

## \_\_init\_\_

\_\_init\_\_(  
    capacity,  
    dtypes,  
    shapes=None,  
    names=None,  
    shared\_name=None,  
    name='fifo\_queue'  
)

Creates a queue that dequeues elements in a first-in first-out order.

A FIFOQueue has bounded capacity; supports multiple concurrent producers and consumers; and provides exactly-once delivery.

A FIFOQueue holds a list of up to capacity elements. Each element is a fixed-length tuple of tensors whose dtypes are described by dtypes, and whose shapes are optionally described by the shapesargument.

If the shapes argument is specified, each component of a queue element must have the respective fixed shape. If it is unspecified, different queue elements may have different shapes, but the use of dequeue\_many is disallowed.

#### Args:

* **capacity**: An integer. The upper bound on the number of elements that may be stored in this queue.
* **dtypes**: A list of DType objects. The length of dtypes must equal the number of tensors in each queue element.
* **shapes**: (Optional.) A list of fully-defined TensorShape objects with the same length as dtypes, or None.
* **names**: (Optional.) A list of string naming the components in the queue with the same length as dtypes, or None. If specified the dequeue methods return a dictionary with the names as keys.
* **shared\_name**: (Optional.) If non-empty, this queue will be shared under the given name across multiple sessions.
* **name**: Optional name for the queue operation.

## Properties

### dtypes

The list of dtypes for each component of a queue element.

### name

The name of the underlying queue.

### names

The list of names for each component of a queue element.

### queue\_ref

The underlying queue reference.

### shapes

The list of shapes for each component of a queue element.

## Methods

### close

close(  
    cancel\_pending\_enqueues=False,  
    name=None  
)

Closes this queue.

This operation signals that no more elements will be enqueued in the given queue. Subsequent enqueue and enqueue\_many operations will fail. Subsequent dequeue and dequeue\_manyoperations will continue to succeed if sufficient elements remain in the queue. Subsequently dequeue and dequeue\_many operations that would otherwise block waiting for more elements (if close hadn't been called) will now fail immediately.

If cancel\_pending\_enqueues is True, all pending requests will also be canceled.

#### Args:

* **cancel\_pending\_enqueues**: (Optional.) A boolean, defaulting to False (described above).
* **name**: A name for the operation (optional).

#### Returns:

The operation that closes the queue.

### dequeue

dequeue(name=None)

Dequeues one element from this queue.

If the queue is empty when this operation executes, it will block until there is an element to dequeue.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue is empty, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session istf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

The tuple of tensors that was dequeued.

### dequeue\_many

dequeue\_many(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. All of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are less than n elements left, then an OutOfRange exception is raised.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue contains fewer than n elements, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session is tf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The list of concatenated tensors that was dequeued.

### dequeue\_up\_to

dequeue\_up\_to(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

**Note** This operation is not supported by all queues. If a queue does not support DequeueUpTo, then a [tf.errors.UnimplementedError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/UnimplementedError) is raised.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. If the queue has not been closed, all of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are more than 0 but fewer than n elements remaining, then instead of raising a [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) like tf.QueueBase.dequeue\_many, less than n elements are returned immediately. If the queue is closed and there are 0 elements left in the queue, then a[tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) is raised just like in dequeue\_many. Otherwise the behavior is identical to dequeue\_many.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The tuple of concatenated tensors that was dequeued.

### enqueue

enqueue(  
    vals,  
    name=None  
)

Enqueues one element to this queue.

If the queue is full when this operation executes, it will block until the element has been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary containing the values to enqueue.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a new tuple of tensors to the queue.

### enqueue\_many

enqueue\_many(  
    vals,  
    name=None  
)

Enqueues zero or more elements to this queue.

This operation slices each component tensor along the 0th dimension to make multiple queue elements. All of the tensors in vals must have the same size in the 0th dimension.

If the queue is full when this operation executes, it will block until all of the elements have been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary from which the queue elements are taken.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a batch of tuples of tensors to the queue.

### from\_list

from\_list(  
    index,  
    queues  
)

Create a queue using the queue reference from queues[index].

#### Args:

* **index**: An integer scalar tensor that determines the input that gets selected.
* **queues**: A list of QueueBase objects.

#### Returns:

A QueueBase object.

#### Raises:

* **TypeError**: When queues is not a list of QueueBase objects, or when the data types of queues are not all the same.

### is\_closed

is\_closed(name=None)

Returns true if queue is closed.

This operation returns true if the queue is closed and false if the queue is open.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

True if the queue is closed and false if the queue is open.

### size

size(name=None)

Compute the number of elements in this queue.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

A scalar tensor containing the number of elements in this queue.

# tf.queue.PaddingFIFOQueue

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PaddingFIFOQueue#top_of_page)
* [Class PaddingFIFOQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PaddingFIFOQueue#class_paddingfifoqueue)
  + [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PaddingFIFOQueue#aliases)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PaddingFIFOQueue#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PaddingFIFOQueue#properties)

## Class PaddingFIFOQueue

A FIFOQueue that supports batching variable-sized tensors by padding.

Inherits From: [QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase)

### Aliases:

* Class tf.compat.v1.PaddingFIFOQueue
* Class tf.compat.v1.io.PaddingFIFOQueue
* Class tf.compat.v1.queue.PaddingFIFOQueue
* Class tf.compat.v2.queue.PaddingFIFOQueue
* Class tf.queue.PaddingFIFOQueue

Defined in [python/ops/data\_flow\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/data_flow_ops.py).

A PaddingFIFOQueue may contain components with dynamic shape, while also supporting dequeue\_many. See the constructor for more details.

See [tf.queue.QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase) for a description of the methods on this class.

## \_\_init\_\_

\_\_init\_\_(  
    capacity,  
    dtypes,  
    shapes,  
    names=None,  
    shared\_name=None,  
    name='padding\_fifo\_queue'  
)

Creates a queue that dequeues elements in a first-in first-out order.

A PaddingFIFOQueue has bounded capacity; supports multiple concurrent producers and consumers; and provides exactly-once delivery.

A PaddingFIFOQueue holds a list of up to capacity elements. Each element is a fixed-length tuple of tensors whose dtypes are described by dtypes, and whose shapes are described by the shapesargument.

The shapes argument must be specified; each component of a queue element must have the respective shape. Shapes of fixed rank but variable size are allowed by setting any shape dimension to None. In this case, the inputs' shape may vary along the given dimension, and dequeue\_many will pad the given dimension with zeros up to the maximum shape of all elements in the given batch.

#### Args:

* **capacity**: An integer. The upper bound on the number of elements that may be stored in this queue.
* **dtypes**: A list of DType objects. The length of dtypes must equal the number of tensors in each queue element.
* **shapes**: A list of TensorShape objects, with the same length as dtypes. Any dimension in the TensorShape containing value None is dynamic and allows values to be enqueued with variable size in that dimension.
* **names**: (Optional.) A list of string naming the components in the queue with the same length as dtypes, or None. If specified the dequeue methods return a dictionary with the names as keys.
* **shared\_name**: (Optional.) If non-empty, this queue will be shared under the given name across multiple sessions.
* **name**: Optional name for the queue operation.

#### Raises:

* **ValueError**: If shapes is not a list of shapes, or the lengths of dtypes and shapes do not match, or if names is specified and the lengths of dtypes and names do not match.

## Properties

### dtypes

The list of dtypes for each component of a queue element.

### name

The name of the underlying queue.

### names

The list of names for each component of a queue element.

### queue\_ref

The underlying queue reference.

### shapes

The list of shapes for each component of a queue element.

## Methods

### close

close(  
    cancel\_pending\_enqueues=False,  
    name=None  
)

Closes this queue.

This operation signals that no more elements will be enqueued in the given queue. Subsequent enqueue and enqueue\_many operations will fail. Subsequent dequeue and dequeue\_manyoperations will continue to succeed if sufficient elements remain in the queue. Subsequently dequeue and dequeue\_many operations that would otherwise block waiting for more elements (if close hadn't been called) will now fail immediately.

If cancel\_pending\_enqueues is True, all pending requests will also be canceled.

#### Args:

* **cancel\_pending\_enqueues**: (Optional.) A boolean, defaulting to False (described above).
* **name**: A name for the operation (optional).

#### Returns:

The operation that closes the queue.

### dequeue

dequeue(name=None)

Dequeues one element from this queue.

If the queue is empty when this operation executes, it will block until there is an element to dequeue.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue is empty, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session istf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

The tuple of tensors that was dequeued.

### dequeue\_many

dequeue\_many(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. All of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are less than n elements left, then an OutOfRange exception is raised.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue contains fewer than n elements, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session is tf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The list of concatenated tensors that was dequeued.

### dequeue\_up\_to

dequeue\_up\_to(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

**Note** This operation is not supported by all queues. If a queue does not support DequeueUpTo, then a [tf.errors.UnimplementedError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/UnimplementedError) is raised.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. If the queue has not been closed, all of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are more than 0 but fewer than n elements remaining, then instead of raising a [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) like tf.QueueBase.dequeue\_many, less than n elements are returned immediately. If the queue is closed and there are 0 elements left in the queue, then a[tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) is raised just like in dequeue\_many. Otherwise the behavior is identical to dequeue\_many.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The tuple of concatenated tensors that was dequeued.

### enqueue

enqueue(  
    vals,  
    name=None  
)

Enqueues one element to this queue.

If the queue is full when this operation executes, it will block until the element has been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary containing the values to enqueue.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a new tuple of tensors to the queue.

### enqueue\_many

enqueue\_many(  
    vals,  
    name=None  
)

Enqueues zero or more elements to this queue.

This operation slices each component tensor along the 0th dimension to make multiple queue elements. All of the tensors in vals must have the same size in the 0th dimension.

If the queue is full when this operation executes, it will block until all of the elements have been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary from which the queue elements are taken.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a batch of tuples of tensors to the queue.

### from\_list

from\_list(  
    index,  
    queues  
)

Create a queue using the queue reference from queues[index].

#### Args:

* **index**: An integer scalar tensor that determines the input that gets selected.
* **queues**: A list of QueueBase objects.

#### Returns:

A QueueBase object.

#### Raises:

* **TypeError**: When queues is not a list of QueueBase objects, or when the data types of queues are not all the same.

### is\_closed

is\_closed(name=None)

Returns true if queue is closed.

This operation returns true if the queue is closed and false if the queue is open.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

True if the queue is closed and false if the queue is open.

### size

size(name=None)

Compute the number of elements in this queue.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

A scalar tensor containing the number of elements in this queue.

# tf.queue.PriorityQueue

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PriorityQueue#top_of_page)
* [Class PriorityQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PriorityQueue#class_priorityqueue)
  + [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PriorityQueue#aliases)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PriorityQueue#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/PriorityQueue#properties)

## Class PriorityQueue

A queue implementation that dequeues elements in prioritized order.

Inherits From: [QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase)

### Aliases:

* Class tf.compat.v1.PriorityQueue
* Class tf.compat.v1.io.PriorityQueue
* Class tf.compat.v1.queue.PriorityQueue
* Class tf.compat.v2.queue.PriorityQueue
* Class tf.queue.PriorityQueue

Defined in [python/ops/data\_flow\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/data_flow_ops.py).

See [tf.queue.QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase) for a description of the methods on this class.

## \_\_init\_\_

\_\_init\_\_(  
    capacity,  
    types,  
    shapes=None,  
    names=None,  
    shared\_name=None,  
    name='priority\_queue'  
)

Creates a queue that dequeues elements in a first-in first-out order.

A PriorityQueue has bounded capacity; supports multiple concurrent producers and consumers; and provides exactly-once delivery.

A PriorityQueue holds a list of up to capacity elements. Each element is a fixed-length tuple of tensors whose dtypes are described by types, and whose shapes are optionally described by the shapes argument.

If the shapes argument is specified, each component of a queue element must have the respective fixed shape. If it is unspecified, different queue elements may have different shapes, but the use of dequeue\_many is disallowed.

Enqueues and Dequeues to the PriorityQueue must include an additional tuple entry at the beginning: the priority. The priority must be an int64 scalar (for enqueue) or an int64 vector (for enqueue\_many).

#### Args:

* **capacity**: An integer. The upper bound on the number of elements that may be stored in this queue.
* **types**: A list of DType objects. The length of types must equal the number of tensors in each queue element, except the first priority element. The first tensor in each element is the priority, which must be type int64.
* **shapes**: (Optional.) A list of fully-defined TensorShape objects, with the same length as types, or None.
* **names**: (Optional.) A list of strings naming the components in the queue with the same length as dtypes, or None. If specified, the dequeue methods return a dictionary with the names as keys.
* **shared\_name**: (Optional.) If non-empty, this queue will be shared under the given name across multiple sessions.
* **name**: Optional name for the queue operation.

## Properties

### dtypes

The list of dtypes for each component of a queue element.

### name

The name of the underlying queue.

### names

The list of names for each component of a queue element.

### queue\_ref

The underlying queue reference.

### shapes

The list of shapes for each component of a queue element.

## Methods

### close

close(  
    cancel\_pending\_enqueues=False,  
    name=None  
)

Closes this queue.

This operation signals that no more elements will be enqueued in the given queue. Subsequent enqueue and enqueue\_many operations will fail. Subsequent dequeue and dequeue\_manyoperations will continue to succeed if sufficient elements remain in the queue. Subsequently dequeue and dequeue\_many operations that would otherwise block waiting for more elements (if close hadn't been called) will now fail immediately.

If cancel\_pending\_enqueues is True, all pending requests will also be canceled.

#### Args:

* **cancel\_pending\_enqueues**: (Optional.) A boolean, defaulting to False (described above).
* **name**: A name for the operation (optional).

#### Returns:

The operation that closes the queue.

### dequeue

dequeue(name=None)

Dequeues one element from this queue.

If the queue is empty when this operation executes, it will block until there is an element to dequeue.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue is empty, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session istf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

The tuple of tensors that was dequeued.

### dequeue\_many

dequeue\_many(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. All of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are less than n elements left, then an OutOfRange exception is raised.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue contains fewer than n elements, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session is tf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The list of concatenated tensors that was dequeued.

### dequeue\_up\_to

dequeue\_up\_to(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

**Note** This operation is not supported by all queues. If a queue does not support DequeueUpTo, then a [tf.errors.UnimplementedError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/UnimplementedError) is raised.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. If the queue has not been closed, all of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are more than 0 but fewer than n elements remaining, then instead of raising a [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) like tf.QueueBase.dequeue\_many, less than n elements are returned immediately. If the queue is closed and there are 0 elements left in the queue, then a[tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) is raised just like in dequeue\_many. Otherwise the behavior is identical to dequeue\_many.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The tuple of concatenated tensors that was dequeued.

### enqueue

enqueue(  
    vals,  
    name=None  
)

Enqueues one element to this queue.

If the queue is full when this operation executes, it will block until the element has been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary containing the values to enqueue.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a new tuple of tensors to the queue.

### enqueue\_many

enqueue\_many(  
    vals,  
    name=None  
)

Enqueues zero or more elements to this queue.

This operation slices each component tensor along the 0th dimension to make multiple queue elements. All of the tensors in vals must have the same size in the 0th dimension.

If the queue is full when this operation executes, it will block until all of the elements have been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary from which the queue elements are taken.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a batch of tuples of tensors to the queue.

### from\_list

from\_list(  
    index,  
    queues  
)

Create a queue using the queue reference from queues[index].

#### Args:

* **index**: An integer scalar tensor that determines the input that gets selected.
* **queues**: A list of QueueBase objects.

#### Returns:

A QueueBase object.

#### Raises:

* **TypeError**: When queues is not a list of QueueBase objects, or when the data types of queues are not all the same.

### is\_closed

is\_closed(name=None)

Returns true if queue is closed.

This operation returns true if the queue is closed and false if the queue is open.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

True if the queue is closed and false if the queue is open.

### size

size(name=None)

Compute the number of elements in this queue.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

A scalar tensor containing the number of elements in this queue.

# tf.queue.QueueBase

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase#top_of_page)
* [Class QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase#class_queuebase)
  + [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase#aliases)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase#properties)

## Class QueueBase

Base class for queue implementations.

### Aliases:

* Class tf.compat.v1.QueueBase
* Class tf.compat.v1.io.QueueBase
* Class tf.compat.v1.queue.QueueBase
* Class tf.compat.v2.queue.QueueBase
* Class tf.queue.QueueBase

Defined in [python/ops/data\_flow\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/data_flow_ops.py).

A queue is a TensorFlow data structure that stores tensors across multiple steps, and exposes operations that enqueue and dequeue tensors.

Each queue element is a tuple of one or more tensors, where each tuple component has a static dtype, and may have a static shape. The queue implementations support versions of enqueue and dequeue that handle single elements, versions that support enqueuing and dequeuing a batch of elements at once.

See [tf.queue.FIFOQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/FIFOQueue) and [tf.queue.RandomShuffleQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue) for concrete implementations of this class, and instructions on how to create them.

## \_\_init\_\_

\_\_init\_\_(  
    dtypes,  
    shapes,  
    names,  
    queue\_ref  
)

Constructs a queue object from a queue reference.

The two optional lists, shapes and names, must be of the same length as dtypes if provided. The values at a given index i indicate the shape and name to use for the corresponding queue component in dtypes.

#### Args:

* **dtypes**: A list of types. The length of dtypes must equal the number of tensors in each element.
* **shapes**: Constraints on the shapes of tensors in an element: A list of shape tuples or None. This list is the same length as dtypes. If the shape of any tensors in the element are constrained, all must be; shapes can be None if the shapes should not be constrained.
* **names**: Optional list of names. If provided, the enqueue() and dequeue() methods will use dictionaries with these names as keys. Must be None or a list or tuple of the same length as dtypes.
* **queue\_ref**: The queue reference, i.e. the output of the queue op.

#### Raises:

* **ValueError**: If one of the arguments is invalid.

## Properties

### dtypes

The list of dtypes for each component of a queue element.

### name

The name of the underlying queue.

### names

The list of names for each component of a queue element.

### queue\_ref

The underlying queue reference.

### shapes

The list of shapes for each component of a queue element.

## Methods

### close

close(  
    cancel\_pending\_enqueues=False,  
    name=None  
)

Closes this queue.

This operation signals that no more elements will be enqueued in the given queue. Subsequent enqueue and enqueue\_many operations will fail. Subsequent dequeue and dequeue\_manyoperations will continue to succeed if sufficient elements remain in the queue. Subsequently dequeue and dequeue\_many operations that would otherwise block waiting for more elements (if close hadn't been called) will now fail immediately.

If cancel\_pending\_enqueues is True, all pending requests will also be canceled.

#### Args:

* **cancel\_pending\_enqueues**: (Optional.) A boolean, defaulting to False (described above).
* **name**: A name for the operation (optional).

#### Returns:

The operation that closes the queue.

### dequeue

dequeue(name=None)

Dequeues one element from this queue.

If the queue is empty when this operation executes, it will block until there is an element to dequeue.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue is empty, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session istf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

The tuple of tensors that was dequeued.

### dequeue\_many

dequeue\_many(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. All of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are less than n elements left, then an OutOfRange exception is raised.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue contains fewer than n elements, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session is tf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The list of concatenated tensors that was dequeued.

### dequeue\_up\_to

dequeue\_up\_to(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

**Note** This operation is not supported by all queues. If a queue does not support DequeueUpTo, then a [tf.errors.UnimplementedError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/UnimplementedError) is raised.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. If the queue has not been closed, all of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are more than 0 but fewer than n elements remaining, then instead of raising a [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) like tf.QueueBase.dequeue\_many, less than n elements are returned immediately. If the queue is closed and there are 0 elements left in the queue, then a[tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) is raised just like in dequeue\_many. Otherwise the behavior is identical to dequeue\_many.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The tuple of concatenated tensors that was dequeued.

### enqueue

enqueue(  
    vals,  
    name=None  
)

Enqueues one element to this queue.

If the queue is full when this operation executes, it will block until the element has been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary containing the values to enqueue.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a new tuple of tensors to the queue.

### enqueue\_many

enqueue\_many(  
    vals,  
    name=None  
)

Enqueues zero or more elements to this queue.

This operation slices each component tensor along the 0th dimension to make multiple queue elements. All of the tensors in vals must have the same size in the 0th dimension.

If the queue is full when this operation executes, it will block until all of the elements have been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary from which the queue elements are taken.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a batch of tuples of tensors to the queue.

### from\_list

@staticmethod  
from\_list(  
    index,  
    queues  
)

Create a queue using the queue reference from queues[index].

#### Args:

* **index**: An integer scalar tensor that determines the input that gets selected.
* **queues**: A list of QueueBase objects.

#### Returns:

A QueueBase object.

#### Raises:

* **TypeError**: When queues is not a list of QueueBase objects, or when the data types of queues are not all the same.

### is\_closed

is\_closed(name=None)

Returns true if queue is closed.

This operation returns true if the queue is closed and false if the queue is open.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

True if the queue is closed and false if the queue is open.

### size

size(name=None)

Compute the number of elements in this queue.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

A scalar tensor containing the number of elements in this queue.

# tf.queue.RandomShuffleQueue

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue#top_of_page)
* [Class RandomShuffleQueue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue#class_randomshufflequeue)
  + [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue#aliases)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/RandomShuffleQueue#properties)

## Class RandomShuffleQueue

A queue implementation that dequeues elements in a random order.

Inherits From: [QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase)

### Aliases:

* Class tf.compat.v1.RandomShuffleQueue
* Class tf.compat.v1.io.RandomShuffleQueue
* Class tf.compat.v1.queue.RandomShuffleQueue
* Class tf.compat.v2.queue.RandomShuffleQueue
* Class tf.queue.RandomShuffleQueue

Defined in [python/ops/data\_flow\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/data_flow_ops.py).

See [tf.queue.QueueBase](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/queue/QueueBase) for a description of the methods on this class.

## \_\_init\_\_

\_\_init\_\_(  
    capacity,  
    min\_after\_dequeue,  
    dtypes,  
    shapes=None,  
    names=None,  
    seed=None,  
    shared\_name=None,  
    name='random\_shuffle\_queue'  
)

Create a queue that dequeues elements in a random order.

A RandomShuffleQueue has bounded capacity; supports multiple concurrent producers and consumers; and provides exactly-once delivery.

A RandomShuffleQueue holds a list of up to capacity elements. Each element is a fixed-length tuple of tensors whose dtypes are described by dtypes, and whose shapes are optionally described by the shapes argument.

If the shapes argument is specified, each component of a queue element must have the respective fixed shape. If it is unspecified, different queue elements may have different shapes, but the use of dequeue\_many is disallowed.

The min\_after\_dequeue argument allows the caller to specify a minimum number of elements that will remain in the queue after a dequeue or dequeue\_many operation completes, to ensure a minimum level of mixing of elements. This invariant is maintained by blocking those operations until sufficient elements have been enqueued. The min\_after\_dequeue argument is ignored after the queue has been closed.

#### Args:

* **capacity**: An integer. The upper bound on the number of elements that may be stored in this queue.
* **min\_after\_dequeue**: An integer (described above).
* **dtypes**: A list of DType objects. The length of dtypes must equal the number of tensors in each queue element.
* **shapes**: (Optional.) A list of fully-defined TensorShape objects with the same length as dtypes, or None.
* **names**: (Optional.) A list of string naming the components in the queue with the same length as dtypes, or None. If specified the dequeue methods return a dictionary with the names as keys.
* **seed**: A Python integer. Used to create a random seed. See [tf.compat.v1.set\_random\_seed](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/set_random_seed)for behavior.
* **shared\_name**: (Optional.) If non-empty, this queue will be shared under the given name across multiple sessions.
* **name**: Optional name for the queue operation.

## Properties

### dtypes

The list of dtypes for each component of a queue element.

### name

The name of the underlying queue.

### names

The list of names for each component of a queue element.

### queue\_ref

The underlying queue reference.

### shapes

The list of shapes for each component of a queue element.

## Methods

### close

close(  
    cancel\_pending\_enqueues=False,  
    name=None  
)

Closes this queue.

This operation signals that no more elements will be enqueued in the given queue. Subsequent enqueue and enqueue\_many operations will fail. Subsequent dequeue and dequeue\_manyoperations will continue to succeed if sufficient elements remain in the queue. Subsequently dequeue and dequeue\_many operations that would otherwise block waiting for more elements (if close hadn't been called) will now fail immediately.

If cancel\_pending\_enqueues is True, all pending requests will also be canceled.

#### Args:

* **cancel\_pending\_enqueues**: (Optional.) A boolean, defaulting to False (described above).
* **name**: A name for the operation (optional).

#### Returns:

The operation that closes the queue.

### dequeue

dequeue(name=None)

Dequeues one element from this queue.

If the queue is empty when this operation executes, it will block until there is an element to dequeue.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue is empty, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session istf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

The tuple of tensors that was dequeued.

### dequeue\_many

dequeue\_many(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. All of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are less than n elements left, then an OutOfRange exception is raised.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed, the queue contains fewer than n elements, and there are no pending enqueue operations that can fulfill this request, [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) will be raised. If the session is tf.Session.close, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The list of concatenated tensors that was dequeued.

### dequeue\_up\_to

dequeue\_up\_to(  
    n,  
    name=None  
)

Dequeues and concatenates n elements from this queue.

**Note** This operation is not supported by all queues. If a queue does not support DequeueUpTo, then a [tf.errors.UnimplementedError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/UnimplementedError) is raised.

This operation concatenates queue-element component tensors along the 0th dimension to make a single component tensor. If the queue has not been closed, all of the components in the dequeued tuple will have size n in the 0th dimension.

If the queue is closed and there are more than 0 but fewer than n elements remaining, then instead of raising a [tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) like tf.QueueBase.dequeue\_many, less than n elements are returned immediately. If the queue is closed and there are 0 elements left in the queue, then a[tf.errors.OutOfRangeError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/OutOfRangeError) is raised just like in dequeue\_many. Otherwise the behavior is identical to dequeue\_many.

#### Args:

* **n**: A scalar Tensor containing the number of elements to dequeue.
* **name**: A name for the operation (optional).

#### Returns:

The tuple of concatenated tensors that was dequeued.

### enqueue

enqueue(  
    vals,  
    name=None  
)

Enqueues one element to this queue.

If the queue is full when this operation executes, it will block until the element has been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary containing the values to enqueue.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a new tuple of tensors to the queue.

### enqueue\_many

enqueue\_many(  
    vals,  
    name=None  
)

Enqueues zero or more elements to this queue.

This operation slices each component tensor along the 0th dimension to make multiple queue elements. All of the tensors in vals must have the same size in the 0th dimension.

If the queue is full when this operation executes, it will block until all of the elements have been enqueued.

At runtime, this operation may raise an error if the queue is tf.QueueBase.close before or during its execution. If the queue is closed before this operation runs, [tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised. If this operation is blocked, and either (i) the queue is closed by a close operation with cancel\_pending\_enqueues=True, or (ii) the session is tf.Session.close,[tf.errors.CancelledError](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/errors/CancelledError) will be raised.

#### Args:

* **vals**: A tensor, a list or tuple of tensors, or a dictionary from which the queue elements are taken.
* **name**: A name for the operation (optional).

#### Returns:

The operation that enqueues a batch of tuples of tensors to the queue.

### from\_list

from\_list(  
    index,  
    queues  
)

Create a queue using the queue reference from queues[index].

#### Args:

* **index**: An integer scalar tensor that determines the input that gets selected.
* **queues**: A list of QueueBase objects.

#### Returns:

A QueueBase object.

#### Raises:

* **TypeError**: When queues is not a list of QueueBase objects, or when the data types of queues are not all the same.

### is\_closed

is\_closed(name=None)

Returns true if queue is closed.

This operation returns true if the queue is closed and false if the queue is open.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

True if the queue is closed and false if the queue is open.

### size

size(name=None)

Compute the number of elements in this queue.

#### Args:

* **name**: A name for the operation (optional).

#### Returns:

A scalar tensor containing the number of elements in this queue.