PriorityBlockingQueue Class in Java

**PriorityBlockingQueue** is an unbounded blocking queue that uses the same ordering rules as class **[PriorityQueue](https://www.geeksforgeeks.org/priority-queue-class-in-java-2/)** and supplies blocking retrieval operations. Since it is unbounded, adding elements may sometimes fail due to resource exhaustion resulting in **[OutOfMemoryError](https://www.geeksforgeeks.org/understanding-outofmemoryerror-exception-java/)**. This class does not permit null elements.

PriorityBlockingQueue class and its iterator implements all of the optional methods of the Collection and Iterator interfaces. The Iterator provided in method iterator() is not guaranteed to traverse the elements of the PriorityBlockingQueue in any particular order. For ordered traversal, use **[Arrays.sort(pq.toArray())](https://www.geeksforgeeks.org/arrays-sort-in-java-with-examples/)**. Also, method drainTo() can be used to remove some or all elements in priority order and place them in another collection.

Operations on this class make no guarantees about the ordering of elements with equal priority. If an ordering is needed to be enforced, define custom classes or comparators that use a secondary key to break ties in primary priority values.

This class is a member of the Java Collections Framework.

**Class Heirarchy:**

java.lang.Object

↳ java.util.AbstractCollection

↳ java.util.AbstractQueue

↳ java.util.concurrent.PriorityBlockingQueue

**Type Parameters:** The type parameter of PriorityBlockingQueue **E** is the type of elements held in this collection

**Implemented Interfaces:** Following are the interfaces implemented by the PriorityBlockingQueue Class

* Serializable
* Iterable
* Collection
* BlockingQueue
* Queue

**Syntax:**

public class PriorityBlockingQueue

extends AbstractQueue

implements BlockingQueue, Serializable

**Constructor Summary:**

1. **PriorityBlockingQueue()**  
   Creates a PriorityBlockingQueue with the default initial capacity (11) that orders its elements according to their natural ordering. Adding element more than the initial capacity changes the capacity of the PriorityBlockingQueue dynamically as the PriorityBlockingQueue is not capacity constrained.