▲

Languages

System.out.println in Java

How to calculate log base 2 of an Integer in Java?

Java Ternary Operator with Examples

How to find duplicate elements in a Stream in Java

How to iterate over a TreeMap in Java?

How to create a COVID-19 Tracker Android App

Remove first element from ArrayList in Java

Removing last element from ArrayList in Java

How to set Precision for Double values in Java?

Java IO : Input-output in Java with Examples

Monolithic vs Microservices architecture

How to validate an IP address using Regular Expressions in Java

What Will Be The Best Java IDE's in 2020?

Java program to print Even length words in a String

In Java, Can we call the main() method of a class from another class?

Difference between Increment and Decrement Operators

Program to check if a String in Java contains only whitespaces

Difference between Core Java and Advanced Java

How to iterate over a 2D list (list of lists) in Java

Program to check if the String is Null in Java

How to pad a String in Java

▲

Queue Interface In Java

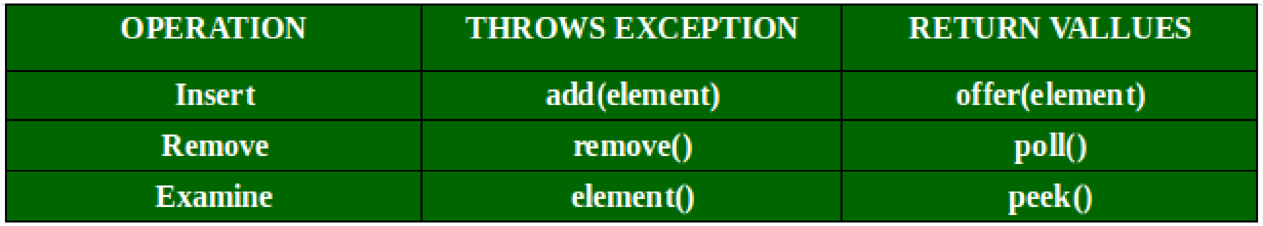
The Queue interface is available in java.util package and extends the Collection interface. The queue collection is used to hold the elements about to be processed and provides various operations like the insertion, removal etc. It is an ordered list of objects with its use limited to insert elements at the end of the list and deleting elements from the start of list i.e. it follows the FIFO or the First-In-First-Out principle. Being an interface the queue needs a concrete class for the declaration and the most common classes are the PriorityQueue and LinkedList in Java.It is to be noted that both the implementations are not thread safe. *PriorityBlockingQueue* is one alternative implementation if thread safe implementation is needed. Few important characteristics of Queue are:

- The Queue is used to insert elements at the end of the queue and removes from the beginning of the queue. It follows FIFO concept.
- The Java Queue supports all methods of Collection interface including insertion, deletion etc.
- LinkedList, ArrayBlockingQueue and PriorityQueue are the most frequently used implementations.
- If any null operation is performed on BlockingQueues, NullPointerException is thrown.

- BlockingQueues have thread-safe implementations.
- The Queues which are available in java.util package are Unbounded Queues
- The Queues which are available in java.util.concurrent package are the Bounded Queues.
- All Queues except the Deques supports insertion and removal at the tail and head of the queue respectively. The Deques support element insertion and removal at both ends.

Methods in Queue:

1. **add()-** This method is used to add elements at the tail of queue. More specifically, at the last of linked-list if it is used, or according to the priority in case of priority queue implementation.
2. **peek()-** This method is used to view the head of queue without removing it. It returns Null if the queue is empty.
3. **element()-** This method is similar to peek(). It throws *NoSuchElementException* when the queue is empty.
4. **remove()-** This method removes and returns the head of the queue. It throws *NoSuchElementException* when the queue is empty.
5. **poll()-** This method removes and returns the head of the queue. It returns null if the queue is empty.
6. **size()-** This method return the no. of elements in the queue.



OPERATION	THROWS EXCEPTION	RETURN VALLUES
Insert	add(element)	offer(element)
Remove	remove()	poll()
Examine	element()	peek()

Since it is a subtype of Collections class, it inherits all the methods of it namely *size(), isEmpty(), contains() etc.*

Below is a simple Java program to demonstrate these methods:

```
// Java orogram to demonstrate working of Queue
// interface in Java
import java.util.LinkedList;
import java.util.Queue;

public class QueueExample
{
  public static void main(String[] args)
  {
    Queue<Integer> q = new LinkedList<>();

    // Adds elements {0, 1, 2, 3, 4} to queue
    for (int i=0; i<5; i++)
     q.add(i);

    // Display contents of the queue.
    System.out.println("Elements of queue-"+q);

    // To remove the head of queue.
    int removedele = q.remove();
    System.out.println("removed element-" + removedele);

    System.out.println(q);

    // To view the head of queue
    int head = q.peek();
    System.out.println("head of queue-" + head);

    // Rest all methods of collection interface,
    // Like size and contains can be used with this
    // implementation.
    int size = q.size();
    System.out.println("Size of queue-" + size);
  }
}
```

Output:

```
Elements of queue-[0, 1, 2, 3, 4]
removed element-0
[1, 2, 3, 4]
head of queue-1
Size of queue-4
```

Applications of queue data structure can be found here

Recommended Posts:

Java.util.function.BiPredicate interface in Java with Examples

Java.util.function.LongPredicate interface in Java with Examples

Java.util.function.DoublePredicate interface in Java with Examples

Java.util.function.IntPredicate interface in Java with Examples

Java 8 | DoubleToLongFunction Interface in Java with Examples

Java 8 | Consumer Interface in Java with Examples

Java 8 | BiConsumer Interface in Java with Examples

Java 8 | IntToDoubleFunction Interface in Java with Examples

Map Interface in Java

Runnable interface in Java

Nested Interface in Java

Marker interface in Java

Java 8 | ObjDoubleConsumer Interface with Example

IntUnaryOperator Interface in Java

DoubleUnaryOperator Interface in Java

Improved By : Chinmoy Lenka, BharatGupta3, codekaust

Article Tags : Java Queue Java-Collections java-queue

Practice Tags : Java Queue Java-Collections

26

1.4

☐ To-do ☐ Done

Based on **35** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

▲

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

▲