Queue Interface In Java

The java.util.Queue is a subtype of java.util.Collection interface. It is an ordered list of objects with its use limited to inserting elements at the end of list and deleting elements from the start of list i.e. it follows FIFO principle.

Since it is an interface, we need a concrete class during its declaration. There are many ways to initialize a Queue object, most common being-

1. As a Priority Queue
2. As a LinkedList

Please note that both the implementations are not thread safe. PriorityBlockingQueue is one alternative implementation if you need a thread safe implementation.

Operations on Queue :

- **Add()-**Adds an element at the tail of queue. More specifically, at the last of linkedlist if it is used, or according to the priority in case of priority queue implementation.
- **peek()-**To view the head of queue without removing it. Returns null if queue is empty.
- **element()-**Similar to peek(). Throws NoSuchElementException if queue is empty.
- **remove()-**Removes and returns the head of the queue. Throws NoSuchElementException when queue is impty.
- **poll()-**Removes and returns the head of the queue. Returns null if queue is empty.

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

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);
  }
}
```

Run on IDE

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

This article is contributed by **Rishabh Mahrsee** .If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GATE CS Corner   Company Wise Coding Practice

Java   Java-Collections