Difference between poll() and remove() in Queue

[Queue<E>](http://data-structure-learning.blogspot.com/2015/05/queue-interface.html) is one of the [interfaces](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-2-interfaces.html) in [Java Collections Framework](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-1.html). It is recommended to study the Queue<E> interface before we dive deeper in the difference of poll() and remove() method of Queue<E> interface.

|  |  |
| --- | --- |
| E poll(); | Retrieves and removes head of this queue.  Returns null if Queue is empty. |
| E remove(); | Retrieves and removes head of this queue. Throws Exception (NoSuchElementException) if Queue is empty. |

Below is the code snippet take from [AbstactQueue<E>](http://data-structure-learning.blogspot.com/2015/05/abstractqueue-class.html) class.

**public** E remove() {

E x = poll();

**if** (x != **null**)

**return** x;

**else**

**throw** **new** NoSuchElementException();

}

remove() method calls the poll() method. If poll() returns null then NoSuchElementException() exception is thrown. Else the element is retrieved.

Caution: It is very tempting to assume that poll() or remove() are in Queue<E> interface they will return the front element. No they do not, not in all cases. Remember Queue<E> is interface and it holds elements prior to processing. So the ordering responsibility of elements is on concrete classes that implement Queue<E> interface.