Queue, PriorityQueue源码分析

笔记本: DSA
创建时间: 2020/8/29 18:49 **更新时间:** 2020/8/29 23:14
作者: Jie Zhong
URL: http://developer.classpath.org/doc/java/util/PriorityQueue-source.html

---

java.util.Queue 接口
常见的实现类
LinkedList, PriorityQueue

Summary of Queue methods

	Throw exception	Returns Special value
Insert	add(e)	offer(e)
Remove	remove(e)	poll(e)
Examine	element(e)	peek(e)

示例代码

```
Queue<String> queue = new  LinkedList<String>();
queue.offer("one");
queue.offer("two");
queue.offer("three");
queue.offer("four");
System.out.println(queue);

String  polledElement = queue.poll();
System.out.println(polledElement);
System.out.println(queue);

String  peekedElement = queue.peek();
System.out.println(peekedElement);
System.out.println(queue);

while(queue.size() > 0) {
    System.out.println(queue.poll());
}
```

实现类PriorityQueue
通过堆来实现，以及比较器 Comparator实现优先级比较
构造函数 PriorityQueue(Comparator<? super E> comparator)

方法	

boolean offer(E o)	public boolean offer(E o) {
	if (o == null)
	throw new NullPointerException;
	
	int slot = findSlot(-1);

	storage[slot] = o;
	++used;
	bubbleUp(slot); //调整堆，保持堆的性质
	
	return true；
	}

E poll()	public E poll() {
	if (used == 0)
	reurn null;
	E result = storage[0];
	remove(0);
	return result;
	}

E peek()	public E peek() {
	return used == 0 ? null : storage[0];
	}

实现类LinkedList

通过双向链表来实现

方法		方法	

boolean
add(T o)
	public boolean add(T o) {
	addLastEntry(new
	Entry<T>(o));
	return true;
	}
	在链表尾部增加结点
		boolean
		offer(T o)
			public offer(T value) {
			return add(value);
			}

T remove()	public T remove() {
	return removeFirst();
	}
	删除链表的头结点，并返回该结点
		T poll()	public T poll() {
			if (size == 0)
			return null;
			return removeFirst();
			}

T element()	public T elemet() {
	return getFirst();
	}
	获得链表的头结点
		T peek()	public T peek() {
			if (size == 0)
			return null;
			return getFirst();
			}