Hello …..,

Thank you for contacting Telerik Support.

I can see your point on the matter. Let me see how we can get this done.

1. First lets **explain** what is the ” dequeue” :

In computer science, **double-ended queue** (“**dequeue”**, often abbreviated to “**deque”**, pronounced **“deck”**) is an abstract data type - a linear collection that supports element insertion and removal at both ends. Basically, the **Deque** combines the functionality of both queue and stack data structures. Said formally, **Deque** generalizes a queue, for which elements can be added to or removed from either the **front** (head) or **back** (tail). Normal **Deque** implementations place no fixed limits on the number of elements they may contain.

1. We can get started with entering the right “**using**” for your implementation of the Dequeue, and it is: using System.Collections.Generic; so that your LinkedList<T> has its reference.
2. After that you enter your public class MyDeque<T>
3. Add a constructor: public MyDeque() { public LinkedList<T> data = new LinkedList<T>(); }
4. After all that you can start to implement your methods and properties for the dequeue which are:

**- int Count** – returns the number of elements in the **Deque**;

**- void PushFirst(T element)** - inserts element at the beginning of the **Deque**;

**- void PushLast(T element)** - inserts element at the end of the **Deque**;

**- T PopFirst()** - removes and returns the element at the beginning of the **Deque**;

**- T PopLast()** - removes and returns the element at the end of the **Deque**;

**- T PeekFirst()** - returns the element at the beginning of the **Deque** without removing it;

**- T PeekLast()** - returns the element at the end of the **Deque** without removing it;

**- void Clear()** - clears the **Deque**. It will be empty after this operation completes;

**- bool Contains(T element)** - returns true if this collection contains the specified element;

For your convenience I have attached an Deque.cs file, to this thread, in which you can see the detailed implementation of all the methods and properties.

Our suggestions and recommendations on the matter are:

I hope you will find the Deque<T> class to be a useful addition to the other collection classes in the .NET Framework.

It is not intended as a replacement for the Queue<T> class and should only be used in situations where additions and removals at both ends of the queue are required. The performance of the two classes should be much the same.

I hope this information helps. Please let me know if I can assist you any further. Thank you in advance.

Regards,

…… ……  
the Telerik team