|  |  |
| --- | --- |
| Internationale Hochschule, DSOPDJK | 03.06.2023 |
| Semester 2 |  |
| Dozent / Lehrer: Herr Klemm |  |
| Referent: Müller Korbinian |  |

**Queues**

Inhalt

[Was ist eine Queue 1](#_Toc167809200)

[Interfaces 2](#_Toc167809201)

[Quellcode 2](#_Toc167809202)

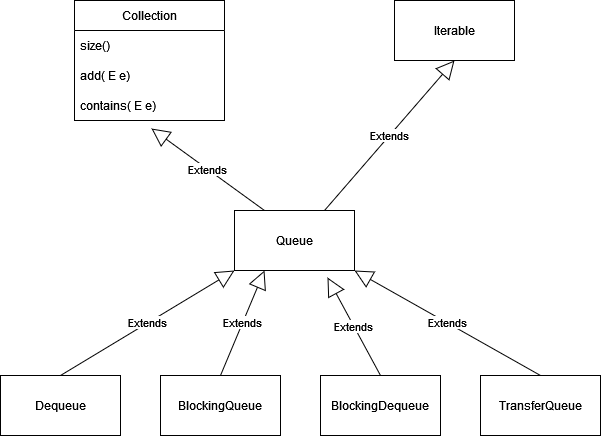
[MyQueue.java 2](#_Toc167809203)

[Node.java 3](#_Toc167809204)

# Was ist eine Queue

* Datenstruktur
* First In First Out -> Zuerst eingefügtes Element wird als ersten ausgegeben
* Priority Queue-> Elemente anhand Wichtigkeit oder andere Vorgabe sortiert
* Dequeue -> Einfügen und entfernen an beiden Seiten möglich
* BlockingQueue -> Nur ein Thread kann auf die Queue zugreifen
* TransferQueue -> Bildet Transaktion

Quellen:  
Queue (Java Platform SE 8 ). (2024, April 4). <https://docs.oracle.com/javase/8/docs/api/java/util/Queue.html>  
  
Collection(Java Platform SE 8 ). (2024, April 4). https://docs.oracle.com/javase/8/docs/api/java/util/Collection.html

Interfaces

Quellcode

## MyQueue.java

package Queue;  
  
public class MyQueue<T>{  
  
 Node<T> head;  
 Node<T> tail;  
  
 public MyQueue(){  
 this.tail = null;  
 this.head = null;  
 }  
  
 public void add (T element){  
 Node<T> newNode = new Node<>();  
 newNode.data = element;  
 try{  
 if(head == null && tail == null){  
 this.head = this.tail = newNode;  
 }else{  
 this.tail.next = newNode;  
 this.tail = newNode;  
 }  
 }catch(Error e){  
 throw new RuntimeException("insert gone wrong");  
 }  
 }  
  
 public boolean offer(T element){  
 Node<T> newNode = new Node<>();  
 newNode.data = element;  
 try{  
 if(head == null && tail == null){  
 this.head = this.tail = newNode;  
 }else{  
 this.tail.next = newNode;  
 this.tail = newNode;  
 }  
 return true;  
 }catch(Error e){  
 return false;  
 }  
 }  
  
 public T remove(){  
  
 try{  
 Node<T> rmElement = this.head;  
 this.head = rmElement.next;  
 return rmElement.data;  
 }catch(Error e){  
 throw new RuntimeException("remove gone wrong");  
 }  
 }  
  
 public T poll(){  
 Node<T> rmElement = this.head;  
 this.head = rmElement.next;  
 return rmElement.data;  
 }  
  
 public T element(){  
 return this.head.data;  
 }  
  
 public T peek(){  
 return this.head.data;  
 }  
}

## Node.java

package Queue;  
  
public class Node<T> {  
 public T data;  
 public Node<T> next;  
  
 public Node(){  
 this.next = null;  
 this.data = null;  
 }  
}