转载请注明链接:https://blog.csdn.net/feather_wch/article/details/82500481

> 详解Collection集合中各个部分的知识点。有部分集合会额外分析源码。

Collection详解

版本号: 2018/9/7-1(18:18)

问题汇总

总览(8)

1、Collection的组成部分

> 1. 具有Collection的接口，根据特性分为 `list`、`Set`、`Queue` 三大类。
> 2. 具有抽象类 `AbstractCollection` 封装了，集合的骨干方法

2、既然已经有了Collection接口，为什么还定义了AbstractCollection这个抽象类?

> 1. List、Set、Queue虽然主体不同，但仍然有共同的行为
> 2. 因此提供 `AbstractCollection` 封装这些行为。

3、Collection相关内容的包

> java.util.*

4、集合的区别

	线程安全性	初始容量	扩容机制	其他
ArrayList	×-异步	0+10(第一次add时)	1.5 * n	
LinkedList	×-异步	不存在	不存在	
SynchronizedList	√-同步	和原始List一致	和原始List一致	
Vector	√-同步	10	2 * n	
Stack	√-同步	10	2 * n	
Hashtable	√-同步	11	2 * n + 1	
HashMap	×-异步	16	2 * n	
ConcurrentHashMap	√-同步	16	2 * n	

> 1. 线程安全的有几种集合：5种
> 2. 非线程安全的有几种集合：3种

5、要求线程安全用什么?

> Vector、hashtable、SynchronizedList、ConcurrentHashMap

6、没有线程安全要求用什么?

> ArrayList、LinkedList、HashMap

7、有键值对?

> HashMap、HashTable

8、数据量很大且要求线程安全

> Vector

Collection(7)

1、Collection接口的作用?

> 1. 集合的根接口

2、Iterable接口的作用?

> 1. 实现该接口能让对象能够成为 `for-each loop` 语句的目标
> 2. iterator()方法：需要返回自己的迭代器

List

3、List接口的作用

> 1. 一种有序集合，实现该接口会具有以下特点：
> 2. 有序
> 3. 可以重复
> 4. 可以有null元素

Set

4、Set接口的作用

> 1. 一种集合，其中的所有元素都不能重复。

SortedSet

5、SortedSet的作用

> 1. 提供有序的set集合(其中所有的元素都不能重复)

Queue

6、Queue接口的作用

> 1. 一种集合，会持有元素的优先级并进行处理。

Deque

7、Deque接口的作用

1. 一种线性集合，支持在两端的元素插入和删除
2. removeFirst()/Last()
3. offerFirst()/Last()
4. pollFirst()/Last()
5. getFirst()/Last()
6. peekFirst()/Last()

AbstractCollection(14)

1、AbstractCollection抽象类的作用

1. implements Collection接口
2. 此类提供了集合接口的骨架实现

AbstractList

2、AbstractList抽象类的作用

1. 继承 `AbstractCollection`
2. 实现 `List` 接口，提供了列表接口的骨架实现。

ArrayList

3、ArrayList的特点(7)

1. `ArrayList` 基于 数组实现(底层是`Object`数组) ,访问元素效率高，插入删除元素的效率低。
2. 实现 `List`接口 ： 意味着 有序、可以重复、可以有`null`元素
3. 实现 `RandomAccess`接口 ： 意味着 `ArrayList` 支持 快速随机访问 ， `Collections` 对于实现了 `RandomAccess`接口 的List会使用 索引二分查找算法 进行查找。
4. 实现 `Cloneable`接口 ： 标识着可以被复制。 `ArrayList` 的 `clone()`复制 其实是 浅复制
5. 实现 `Serializable`接口 ： 标识着集合可以被序列化。
6. 绝大多数情况下应该指定 `ArrayList` 的 初始容量 ： 避免进行 耗时的扩容操作 ， 在能 预知 的情况下尽量使用刚刚好的 列表 ， 从而不浪费任何资源。
7. 非线程安全

4、ArrayList的扩容机制

1. `ArrayList` 的默认容量为 `10` (本质默认初始化为 空数组 ， 第一次 `add` 时会扩容到10)
2. 扩容 以 `1.5`倍 进行扩容。
3. 扩容 ： 会创建一个更大的数组，然后将旧数组的数据都复制过去，该过程是 低效率 的

5、ArrayList源码总结出的效率技巧

> List 提供了 indexOf和lastIndexOf 查询 指定元素的索引(index)，这些方法是从头和从尾部一个个查询的。显然可以直接使用 Collections的binarySearch 通过二分查找，效率会更高。

Vector

6、Vector的特点

> 1. 基于 数组实现
> 2. 支持 快速随机访问
> 3. 线程安全：在不需要 同步时，性能比 ArrayList低，需要同步时使用 SynchronizedList 更好。
> 4. 扩容机制：每次扩容都是原来容量大小的 2倍
> 5. 适合大量数据的时候。

Stack

7、Stack的特点

> 1. Stack 继承自 Vector，扩展了 栈 的相关方法。
> 2. 具有 pop、push、peek 等方法(LinkedList 也有这些方法)。

AbstractSequentialList

8、AbstractSequentialList的作用

> 1. 实现该接口表明需要 sequential access顺序存取 数据，例如，链表(linked list)

LinkedList

9、LinkedList的特点

> 1. LinkedList 基于 链表实现，插入和删除效率高，访问元素效率低(这是链表的特点)
> 2. LinkedList 可以作为 队列、栈 来使用。
> 3. 队列（先进先出）：offerFirst、pollLast
> 4. 栈（后进先出）：pop、push
> 5. 非线程安全

10、LinkedList继承的抽象类和实现的接口?

AbstractSequentialList	List	Deque	Cloneable	Serializable
顺序存储数据	有序、可以重复、元素可以为null	可以两端增加和删除数据	可以拷贝，浅拷贝	可序列化

11、LinkedList实现了几个接口?几个抽象类?

> 1. 4个接口
> 2. 1个抽象类

AbstractSet

12、AbstractSet抽象类的作用

> 1. 继承 `AbstractCollection`
> 2. 实现 `Set` 接口，提供了Set接口的骨架实现。

HashSet

13、HashSet和HashMap的区别与相同

HashMap	HashSet
实现了Map接口	实现Set接口
存储键值对	仅存储对象
调用put（） 向map中添加元素	调用add（）方法向Set中添加元素
HashMap使用键（Key） 计算Hashcode	HashSet使用成员对象来计算hashcode值， 对于两个对象来说hashcode可能相同 (因此使用前要确保重写hashCode（）方法和equals（）方法)
HashMap较快， 因为它是使用唯一的键获取对象	HashSet较慢
非线程安全	非线程安全

LinkedHashSet

TreeSet

AbstractQueue

14、AbstractQueue抽象类的作用

> 1. 继承 `AbstractCollection`

PriorityQueue

ArrayDeque

Collections(2)

SynchronizedList

10、SynchronizedList的特点

> 1. 线程安全
> 2. 通过 `Collections.synchronizedList(new ArrayList<>())`-参数传入任何`List` 就会返回 `SynchronizedList` ， 使得该 `List` 是线程安全的。

SynchronizedMap