集合框架及背后的数据结构

本节目标

> - 了解什么是集合框架
> - 了解学习集合框架的意义
> - 掌握集合框架相关接口和常见的实现类
> - 了解下一阶段要学习的内容

1. 介绍

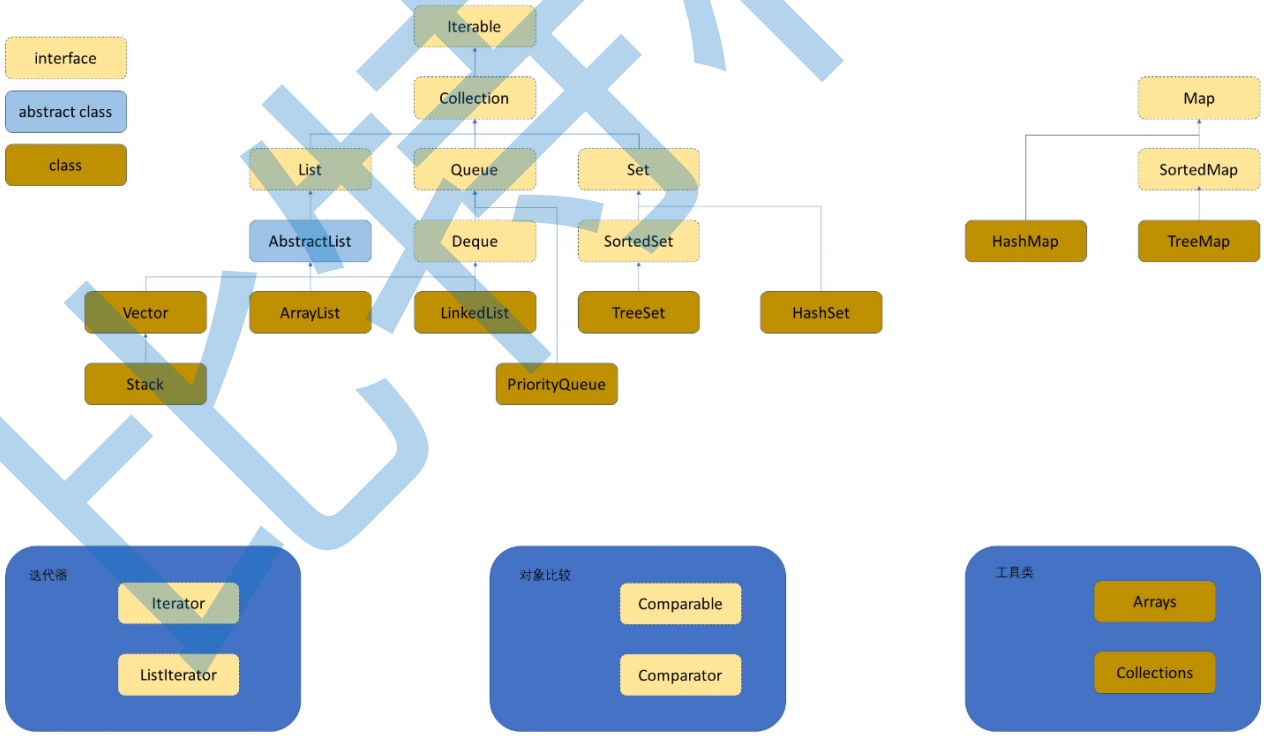
官方教程

Java 集合框架 `Java Collection Framework`，又被称为容器 `container`，是定义在 `java.util` 包下的一组接口 `interfaces` 和其实现类 `classes`。

其主要表现为将多个元素 `element` 置于一个单元中，用于对这些元素进行快速、便捷的存储 `store`、检索 `retrieve`、管理 `manipulate`，即平时我们俗称的增删查改 `CRUD`。

例如，一副扑克牌(一组牌的集合)、一个邮箱(一组邮件的集合)、一个通讯录(一组姓名和电话的映射关系)等等。

类和接口总览



2. 学习的意义

2.1 Java 集合框架的优点及作用

- 使用成熟的集合框架，有助于我们便捷、快速的写出高效、稳定的代码
- 学习背后的数据结构知识，有助于我们理解各个集合的优缺点及使用场景

2.2 笔试及面试题

腾讯-Java后台开发面经

1. HashMap 了解不，介绍一下，如果一个对象为 key 时，hashCode 和 equals 方法的用法要注意什么?
2. HashSet 和 HashMap 的区别是什么?
3. HashMap 是线程安全的么? 那需要线程安全需要用到什么?

阿里巴巴-Java后台开发面经

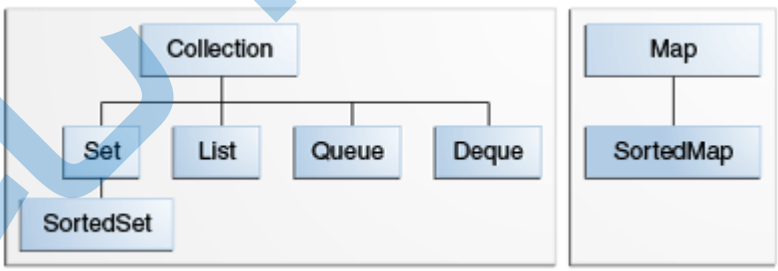
1. ArrayList 和 LinkedList 的区别是什么?
2. 有了解过 HashMap 的具体实现么?
3. HashMap 和 ConcurrentHashMap 哪个效率更高?

今日头条-Java后台开发面经

1. 编程题：判断一个链表是否是一个回文链表。
2. Redis 的 zset 类型对应到 java 语言中大致是什么类型?
3. hashCode 主要是用来做什么用的?

3. 接口 `interfaces`

3.1 基本关系说明



1. `Collection`：用来存储管理一组对象`objects`，这些对象一般被成为元素`elements`
    1. `Set`：元素不能重复，背后隐含着**查找/搜索**的语义
        1. `SortedSet`：一组有序的不能重复的元素
    2. `List`：线性结构
    3. `Queue`：队列
    4. `Deque`：双端队列

2. `Map`：键值对 `Key-Value-Pair`，背后隐含着**查找/搜索**的语义
    1. `SortedMap`：一组有序的键值对

3.2 Collection 接口说明

Collection 官方文档

3.3 Collection 常用方法说明

方法签名	说明
`boolean add(E e)`	将元素 `e` 放入集合中
`void clear()`	删除集合中的所有元素
`boolean isEmpty()`	判断集合是否没有任何元素，俗称空集合
`boolean remove(Object e)`	如果元素 `e` 出现在集合中，删除其中一个
`int size()`	返回集合中的元素个数
`Object[] toArray()`	返回一个装有所有集合中元素的数组

3.4 Collection 示例

```
import java.util.Collection;
import java.util.ArrayList;
import java.util.Arrays;

public class Demo {
    public static void main(String[] args) {
        Collection<String> list = new ArrayList<>();
        System.out.println(list.size());
        System.out.println(list.isEmpty());
        list.add("我");
        list.add("爱");
        list.add("Java");
        System.out.println(list.size());
        System.out.println(list.isEmpty());
        Object[] array = list.toArray();
        System.out.println(Arrays.toString(array));
        for (String s : list) {
            System.out.println(s);
        }
        list.remove("爱");
        for (String s : list) {
            System.out.println(s);
        }
        list.clear();
        System.out.println(list.size());
        System.out.println(list.isEmpty());
    }
```

```
}
```

运行结果：

```
0
true
3
false
[我, 爱, Java]
我
爱
Java
我
Java
0
true
```

3.5 Map 接口说明

Map 官方文档

3.6 Map 常用方法说明

方法签名	说明
`V get(Object k)`	根据指定的 k 查找对应的 v
`V getOrDefault(Object k, V defaultValue)`	根据指定的 k 查找对应的 v，没有找到用默认值代替
`V put(K key, V value)`	将指定的 k-v 放入 Map
`boolean containsKey(Object key)`	判断是否包含 key
`boolean containsValue(Object value)`	判断是否包含 value
`Set<Map.Entry<K, V>> entrySet()`	将所有键值对返回
`boolean isEmpty()`	判断是否为空
`int size()`	返回键值对的数量

3.7 Map 示例

```
import java.util.Map;
import java.util.HashMap;

public class Demo {
    public static void main(String[] args) {
        Map<String, String> map = new HashMap<>();
        System.out.println(map.size());
        System.out.println(map.isEmpty());
```

```
        System.out.println(map.get("作者"));
        System.out.println(map.getOrDefault("作者", "佚名"));
        System.out.println(map.containsKey("作者"));
        System.out.println(map.containsValue("佚名"));
        map.put("作者", "鲁迅");
        map.put("标题", "狂人日记");
        map.put("发表时间", "1918年");
        System.out.println(map.size());
        System.out.println(map.isEmpty());
        System.out.println(map.get("作者"));
        System.out.println(map.getOrDefault("作者", "佚名"));
        System.out.println(map.containsKey("作者"));
        System.out.println(map.containsValue("佚名"));
        for (Map.Entry<String, String> entry : map.entrySet()) {
            System.out.println(entry.getKey());
            System.out.println(entry.getValue());
        }
    }
}
```

运行结果:

```
0
true
null
佚名
false
false
3
false
鲁迅
鲁迅
true
false
作者
鲁迅
发表时间
1918年
标题
狂人日记
```

4. 实现 `classes`

`interface`	顺序表	链表	堆	红黑树	哈希表
`Set`				`TreeSet`	`HashSet`
`List`	`ArrayList`	`LinkedList`			
`Queue`		`LinkedList`	`PriorityQueue`		
`Deque`		`LinkedList`			
`Map`				`TreeMap`	`HashMap`

除此之外，我们还会学习 java 中的栈 `Stack`

5. 下一阶段

5.1 目标

1. 学习集合框架的基本使用
2. 学习基本的数据结构知识
3. 学习七大基于比较的排序算法
4. 学习相关的 java 知识点

5.2 知识点

1. 集合框架的使用
    1. `Collection`
    2. `List`
    3. `ArrayList`
    4. `LinkedList`
    5. `Stack`
    6. `Queue`
    7. `PriorityQueue`
    8. `Deque`
    9. `Set`
    10. `HashSet`
    11. `TreeSet`
    12. `Map`
    13. `HashMap`
    14. `TreeMap`
    15. `Collections`
2. 数据结构的理论及实现
    1. 顺序表
    2. 链表
    3. 栈
    4. 队列
    5. 二叉树
    6. 堆
3. 排序算法
    1. 插入排序
    2. 希尔排序
    3. 选择排序
    4. 堆排序
    5. 冒泡排序
    6. 快速排序
    7. 归并排序
4. Java 语法
    1. 泛型 `Generic`
    2. 自动装箱 `autobox` 和自动拆箱 `autounbox`
    3. `Object` 的 `equals` 方法

4. `Comparable`和`Comparator`接口

内容重点总结

- Java 集合框架中接口、之间的关系及其含义
- Java 集合框架中接口和其各自对应的常见实现类之间的关系
- 下一阶段的主要课程内容

课后作业

- 博客总结：Java 集合框架中接口、之间的关系及其含义
- 博客总结：Java 集合框架中接口和其各自对应的常见实现类之间的关系
- 博客总结：`List`、`Set`和`Map`的至少一个应用场景