

# JAVA 常用 API

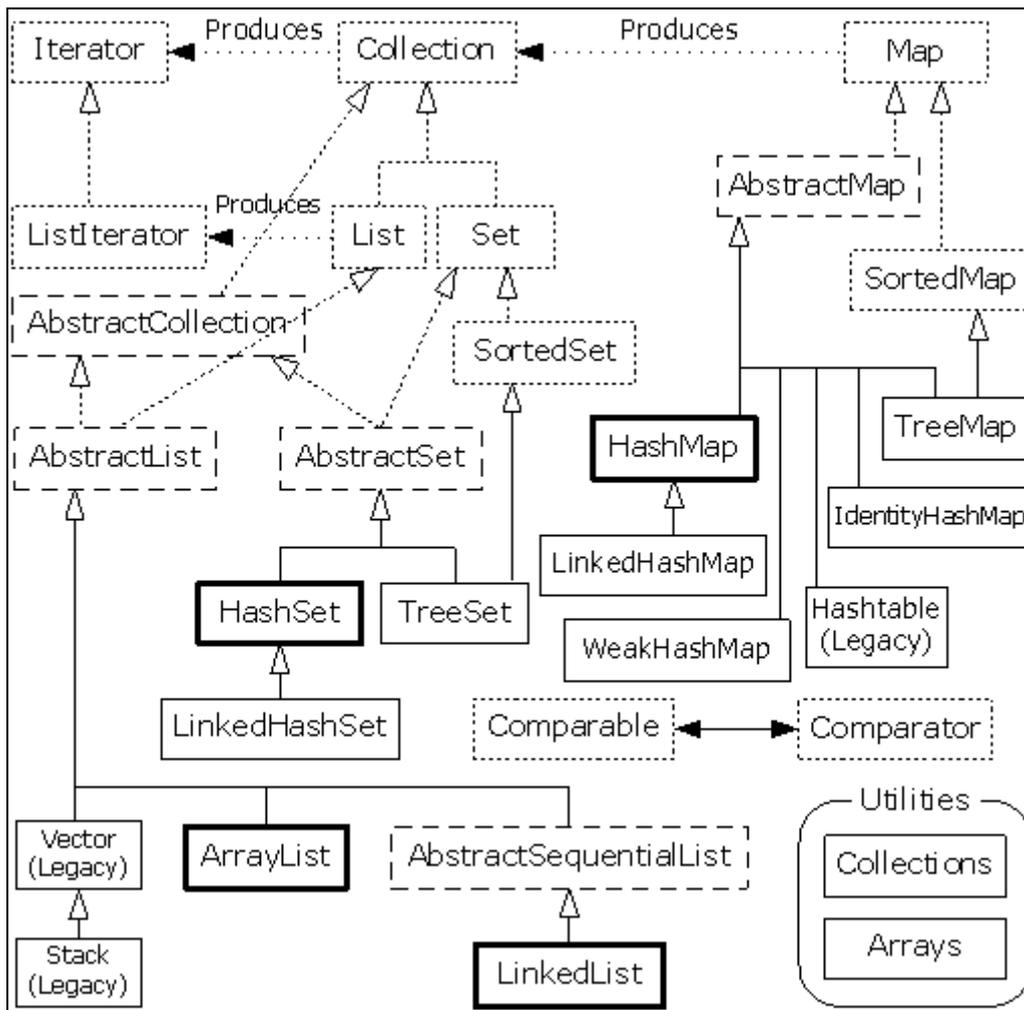
## 1 集合

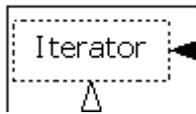
### 1.1 集合就是一个数据的存储容器。

作用：用来代替数组。

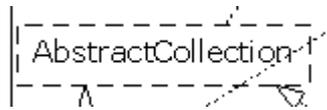
集合是一个可以**改变大小**的存储数据的容器。

### 1.2 集合的整体的框架结构

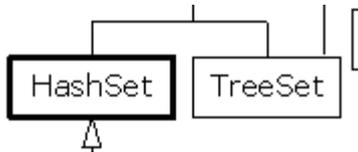




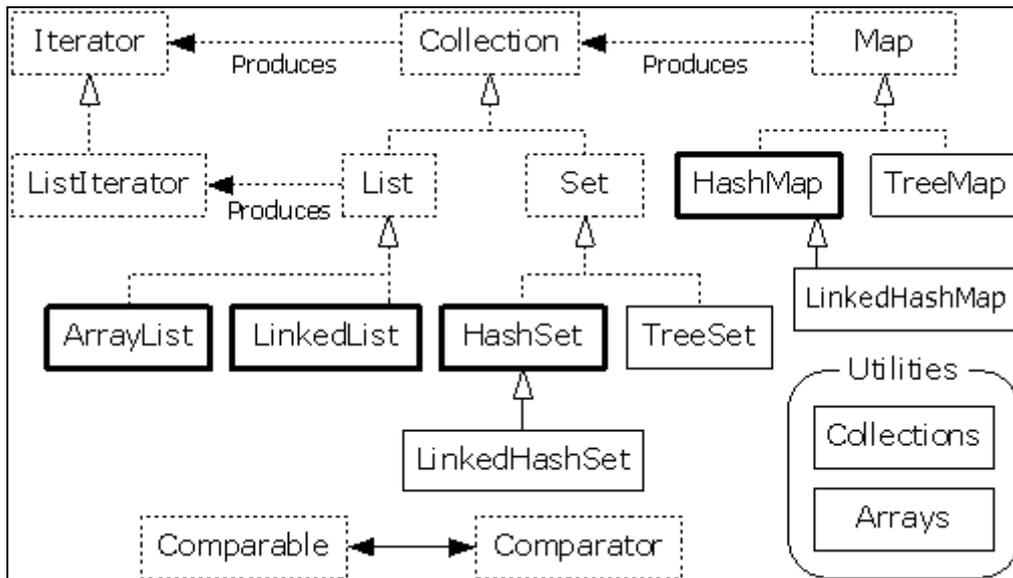
这种使用虚线的是接口。



这种使用点化线的是抽象类。



这种使用实线是类。有加粗是常用的类。



在整个集合框架中顶层的接口：**Collection** 接口。

在 **Collection** 接口下有两个子接口 **List** 和 **Set**

在整个集合框架中还有一个接口 **Map**

**Collections** 操作集合的工具类

**Arrays** 操作数组的工具类

### 1.3 Collection 接口：

线性存储的集合接口。

## 1.4 List 接口

是 `Collection` 接口的子接口，也是线性存储。

### 1.4.1 ArrayList 类

是一种可变大小的数组。

#### 1 创建 `ArrayList` 类的实例。

使用 `new` 关键字。集合对象必须先实例化。

```
ArrayList<String> list = new ArrayList<String>();
```

`<String>` 这个是泛型。用来指明集合中存储的数据的类型。

#### 2 向集合中增加元素。

```
list.add("string");
```

#### 3 从集合中取值。

```
String s = list.get(0);
```

参数 `0` 是索引。

#### 4 迭代集合

Jdk1.5 之前: `for (int i = 0; i < list.size(); i++)` 标准 `for` 循环。淘汰

Jdk1.5 之后: `for (String str : list) forEach` 循环。

## 1.5 Set 接口

是 `Collection` 接口的子接口，也是线性存储。

Set 迭代：

```
public class TestSet {  
    public static void main(String[] args) {  
        Set set = new HashSet();  
        set.add(new String("11"));  
        set.add(new String("222"));  
        Iterator i = set.iterator();// 先迭代出来  
        while (i.hasNext()) {// 遍历  
            System.out.println(i.next());  
        }  
    }  
}
```

## 1.6 Map 接口

是键值映射的集合接口。Key-value

Map 迭代：

方法 1：

```
public static void main(String[] args) {  
    Map map = new HashMap();  
    map.put("1", "one");  
}
```

```
map.put("2", "two");  
map.put("3", "three");  
map.put("4", "four");  
map.put("5", "five");  
map.put("6", "six");  
  
Iterator keys = map.keySet().iterator();// 想取Key
```

形成set

```
while (keys.hasNext()) {  
    String key = (String) keys.next();  
    System.out.println("键" + key + ",值" +  
map.get(key));  
}  
}
```

方法 2:

```
Iterator keys = map.entrySet().iterator();  
  
while(keys.hasNext()) {  
    Map entry =(Map)keys.next();  
    String key=(String)entry.getKey();  
    String value=(String)entry.getValue();  
    System.out.println("键"+key+"="+value);
```

## 2.properties