**Java 集合框架**

早在Java 2中之前，Java就提供了特设类。比如：Dictionary, Vector, Stack, 和Properties这些类用来存储和操作对象组。

虽然这些类都非常有用，但是它们缺少一个核心的，统一的主题。由于这个原因，使用Vector类的方式和使用Properties类的方式有着很大不同。

集合框架被设计成要满足以下几个目标。

* 该框架必须是高性能的。基本集合（动态数组，链表，树，哈希表）的实现也必须是高效的。
* 该框架允许不同类型的集合，以类似的方式工作，具有高度的互操作性。
* 对一个集合的扩展和适应必须是简单的。

为此，整个集合框架就围绕一组标准接口而设计。你可以直接使用这些接口的标准实现，诸如： LinkedList, HashSet, 和 TreeSet等,除此之外你也可以通过这些接口实现自己的集合。

集合框架是一个用来代表和操纵集合的统一架构。所有的集合框架都包含如下内容：

* **接口：**是代表集合的抽象数据类型。接口允许集合独立操纵其代表的细节。在面向对象的语言，接口通常形成一个层次。
* **实现（类）：**是集合接口的具体实现。从本质上讲，它们是可重复使用的数据结构。
* **算法：**是实现集合接口的对象里的方法执行的一些有用的计算，例如：搜索和排序。这些算法被称为多态，那是因为相同的方法可以在相似的接口上有着不同的实现。

除了集合，该框架也定义了几个Map接口和类。Map里存储的是键/值对。尽管Map不是collections，但是它们完全整合在集合中。

**集合接口**

集合框架定义了一些接口。本节提供了每个接口的概述：

|  |  |
| --- | --- |
| **序号** | **接口描述** |
| 1 | Collection 接口 允许你使用一组对象，是Collection层次结构的根接口。 |
| 2 | List 接口 继承于**Collection**和一个 List实例存储一个有序集合的元素。 |
| 3 | Set 继承于 **Collection，是**一个不包含重复元素的集合。 |
| 4 | SortedSet 继承于Set保存有序的集合。 |
| 5 | Map 将唯一的键映射到值。 |
| 6 | Map.Entry 描述在一个Map中的一个元素（键/值对）。是一个Map的内部类。 |
| 7 | SortedMap 继承于Map，使Key保持在升序排列。 |
| 8 | Enumeration 这是一个传统的接口和定义的方法，通过它可以枚举（一次获得一个）对象集合中的元素。这个传统接口已被迭代器取代。 |

**集合类**

Java提供了一套实现了Collection接口的标准集合类。其中一些是具体类，这些类可以直接拿来使用，而另外一些是抽象类，提供了接口的部分实现。

标准集合类汇总于下表：

|  |  |
| --- | --- |
| **序号** | **类描述** |
| 1 | **AbstractCollection** 实现了大部分的集合接口。 |
| 2 | **AbstractList** 继承于AbstractCollection 并且实现了大部分List接口。 |
| 3 | **AbstractSequentialList** 继承于 AbstractList ，提供了对数据元素的链式访问而不是随机访问。 |
| 4 | LinkedList 继承于 AbstractSequentialList，实现了一个链表。 |
| 5 | ArrayList 通过继承AbstractList，实现动态数组。 |
| 6 | **AbstractSet** 继承于AbstractCollection 并且实现了大部分Set接口。 |
| 7 | HashSet 继承了AbstractSet，并且使用一个哈希表。 |
| 8 | LinkedHashSet 具有可预知迭代顺序的 Set 接口的哈希表和链接列表实现。 |
| 9 | TreeSet 继承于AbstractSet，使用元素的自然顺序对元素进行排序. |
| 10 | **AbstractMap** 实现了大部分的Map接口。 |
| 11 | HashMap  继承了HashMap，并且使用一个哈希表。 |
| 12 | TreeMap  继承了AbstractMap，并且使用一颗树。 |
| 13 | WeakHashMap  继承AbstractMap类，使用弱密钥的哈希表。 |
| 14 | LinkedHashMap  继承于HashMap，使用元素的自然顺序对元素进行排序. |
| 15 | IdentityHashMap  继承AbstractMap类，比较文档时使用引用相等。 |

在前面的教程中已经讨论通过java.util包中定义的类，如下所示：

|  |  |
| --- | --- |
| **序号** | **类描述** |
| 1 | Vector  Vector类实现了一个动态数组。和ArrayList和相似，但是两者是不同的。 |
| 2 | Stack  栈是Vector的一个子类，它实现了一个标准的后进先出的栈。 |
| 3 | Dictionary  Dictionary 类是一个抽象类，用来存储键/值对，作用和Map类相似。 |
| 4 | Hashtable  Hashtable是原始的java.util的一部分， 是一个Dictionary具体的实现 。 |
| 5 | Properties  Properties 继承于 Hashtable.表示一个持久的属性集.属性列表中每个键及其对应值都是一个字符串。 |
| 6 | BitSet 一个Bitset类创建一种特殊类型的数组来保存位值。BitSet中数组大小会随需要增加。 |

一个Bitset类创建一种特殊类型的数组来保存位值。BitSet中数组大小会随需要增加。

**集合算法**

集合框架定义了几种算法，可用于集合和映射。这些算法被定义为集合类的静态方法。

在尝试比较不兼容的类型时，一些方法能够抛出 ClassCastException异常。当试图修改一个不可修改的集合时，抛出UnsupportedOperationException异常。

集合定义三个静态的变量：EMPTY\_SET EMPTY\_LIST，EMPTY\_MAP的。这些变量都不可改变。

|  |  |
| --- | --- |
| **序号** | **算法描述** |
| 1 | Collection Algorithms 这里是一个列表中的所有算法实现。 |

**如何使用迭代器**

通常情况下，你会希望遍历一个集合中的元素。例如，显示集合中的每个元素。

做到这一点最简单的方法是采用一个迭代器，它是一个对象，实现了Iterator 接口或ListIterator接口。

迭代器，使你能够通过循环来得到或删除集合的元素。ListIterator继承了Iterator，以允许双向遍历列表和修改元素。

这里通过实例列出Iterator和listIterator接口提供的所有方法。

|  |
| --- |
|  |

**如何使用比较器**

TreeSet和TreeMap的按照排序顺序来存储元素. 然而，这是通过比较器来精确定义按照什么样的排序顺序。

这个接口可以让我们以不同的方式来排序一个集合。

|  |  |
| --- | --- |
| **序号** | **比较器方法描述** |
| 1 | 使用 Java Comparator 这里通过实例列出Comparator接口提供的所有方法 |

**总结**

Java集合框架为程序员提供了预先包装的数据结构和算法来操纵他们。

集合是一个对象，可容纳其他对象的引用。集合接口声明对每一种类型的集合可以执行的操作。

集合框架的类和接口均在java.util包中。