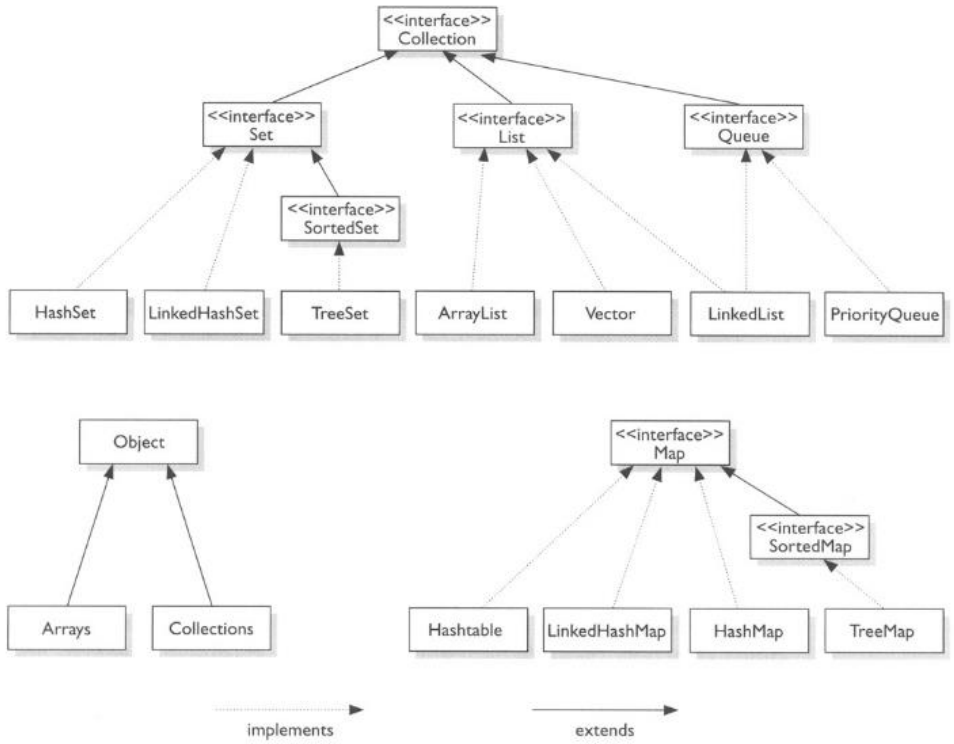
Collection Framework

Collection framework was not part of original Java release. Collections was added to J2SE 1.2.

Collection framework provides many important classes and interfaces to collect and organize group of alike objects.



Advantages of Collection Framework:

1. Reduces programming effort:

   The programmer need not to worry about design of Collection rather than he can focus on its best use in his program.

2. Increases program speed and quality.

3. Reusability and Interoperability

Set Interface :

Set is a collection that cannot contain duplicate elements. This interface models the mathematical set abstraction and is used to represent sets, such as the deck of cards.

The Java platform contains three general-purpose Set implementations: HashSet, TreeSet, and LinkedHashSet. You can use iterator or foreach loop to traverse the elements of a Set.

Set is an interface , so we depend either on HashSet or Treeset

```
import java.util.ArrayList;

import java.util.HashSet;

import java.util.Iterator;

import java.util.List;

import java.util.Set;


public class HashSetExample {

	public static void main(String[] args) {

		Set<String> fruits = new HashSet<>();

		//add example

		fruits.add("Apple");

		fruits.add("Banana");

		//isEmpty example

		System.out.println("fruits set is empty =
"+fruits.isEmpty());

		//contains example

		System.out.println("fruits contains Apple =
"+fruits.contains("Apple"));

		System.out.println("fruits contains Mango =
"+fruits.contains("Mango"));
```

```
            //remove example

            System.out.println("Apple removed from fruits set =
"+fruits.remove("Apple"));

            System.out.println("Mango removed from fruits set =
"+fruits.remove("Mango"));

            //size example

            System.out.println("fruits set size =
"+fruits.size());

            //iterator example

            Iterator<String> iterator = fruits.iterator();

            while(iterator.hasNext()){

            System.out.println("Consuming fruit
"+iterator.next());

            }

      }

}
```

List Interface

List is an ordered collection and can contain duplicate elements. You can access any element from it's index.

List is more like array with dynamic length. List is one of the most used Collection type. ArrayList and LinkedList are implementation classes of List interface.

List interface provides useful methods to add an element at specific index, remove/replace element based on index and to get a sub-list using index.

List Cannot be Instantiated as it is interface.So we either use ArrayList or LinkedList.

```
import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;


public class ArrayToList {

	public static void main(String[] args) {

		String[] vowels = {"a","e","i","o","u"};

		List<String> vowelsList = Arrays.asList(vowels);

		System.out.println(vowelsList);


		List<String> myList = new ArrayList<>();

		for(String s : vowels){

			myList.add(s);

		}

		System.out.println(myList);

		myList.clear();

	}

}
```