ENSF 593/594

5 – Introduction to ArrayLists

A Quick Introduction to Java Collections

What are Collections

- Most programs use collections of data
  - A set of data
  - Words in a dictionary
  - A list of students
  - A list relating people to email addresses
- Java provides a collection framework (JCF), that contains facilities which allow us to represent data as sets, list, and vectors. Also contains useful algorithms that allows the manipulation of the data.

Java Collections

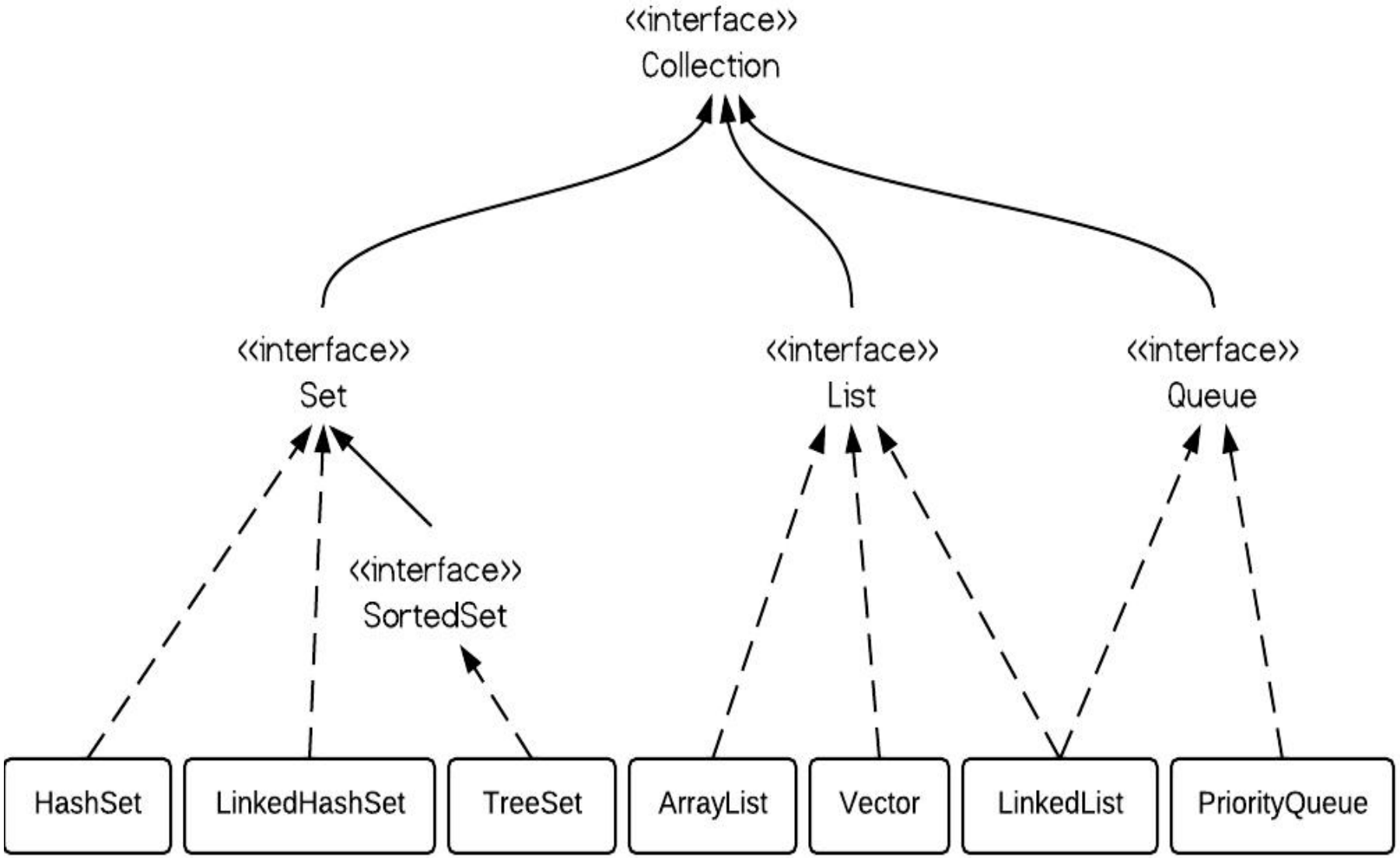

Figure adopted from: http://java.dzone.com/articles/arraylist-vs-linkedlist-vs

Collection

- Collection that can contain duplicate elements, and are in the order that are inserted:
  - ArrayList
  - LinkedList
  - Vector (are thread-safe)
- Collection elements are unique and not necessarily in the order of insertion.
  - HashSet (hash table implementation),
  - TreeSet (a tree data structure. An log(n) time cost operation is guaranteed
- A hash table implemented as a linkedlist. The insertion order is preseved.
  - LinkedHashSet
- Conceptually a Collection that are not ordered, and elements are not unique are called Bag.

Collection <E> in Java

- <E> represents a type, and can be any type other than primitive data types such as int, char, ….
- The Collection public interfaces provides the basis for List-like collections in Java. The interface includes:

```
boolean add(Object)              // Adds a new object to the list
boolean addAll(Collection)       // Adds a collection to the list
void clear()                     // clears the list
boolean contains(Object)         // returns true if list contains an specific object
boolean equals(Object)           // Compares the specified object with this collection
boolean isEmpty()                // returns true if list is empty
Iterator iterator()              // returns an iterator over the elements of the list
boolean remove(Object)           // removes and specific element
boolean removeAll(Collection)    // removes all
int size()                       // returns the number of elements
Object[] toArray()               // returns an array containing all elements
```

Using ArrayList<E> Class

- The following code segment show how to create and use ArrayList objects:

```
ArrayList <Integer> list = new ArrayList <Integer>();
list.add(new Integer(20));
list.add(new Integer(10));
list.add(new Integer(5));
list.add(new Integer(25));

// initialize ArryList b with “list”
ArrayList <Integer> b = new ArrayList(list);
```

Access to the Elements in ArrayList<E>

- Access to the elements of list:

```
list.set(0, 66);
int x = list.get(0);
System.out.println("First element holds: " + x);

// using Iterator object to traverses over the list
Iterator <Interger> i = b.iterator();

while(i.hasNext())
  System.out.println(i.next());
```

- Iterator methods include:
  - hasNext()
  - next() // returns the element
  - remove() // removes the last elemnet

Other Methods

```
// check if list is empty
if(list.isEmpty())
   System.out.println("List is empty and size is " + list.size());

// clear the list
list.clear();

// copy list into an array of object called a
Object []a = list.toArray();
```