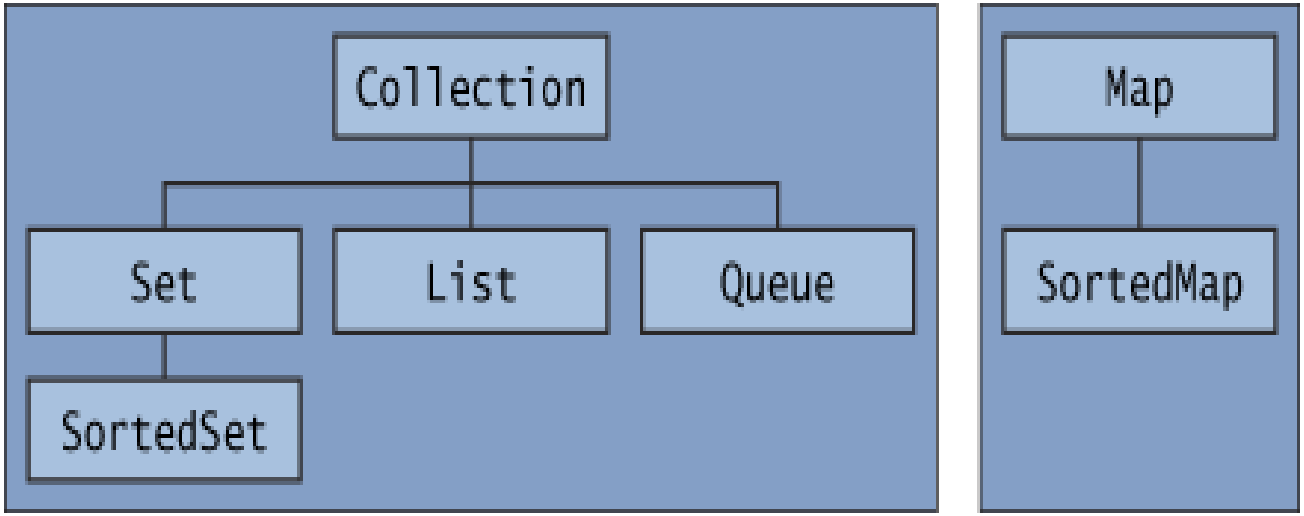
JAVA COLLECTIONS

http://docs.oracle.com/javase/tutorial/collections/index.html

Collection interfaces



public interface Collection<E> extends Iterable<E>

```
public interface Collection<E> extends Iterable<E> {
   // Basic operations
   int size();
   boolean isEmpty();
   boolean contains(Object element);
   boolean add(E element);          //optional
   boolean remove(Object element); //optional
   Iterator<E> iterator();

   // Bulk operations
   boolean containsAll(Collection<?> c);
   boolean addAll(Collection<? extends E> c); //optional
   boolean removeAll(Collection<?> c);        //optional
   boolean retainAll(Collection<?> c);        //optional
   void clear();                              //optional

   // Array operations
   Object[] toArray();
   <T> T[] toArray(T[] a);
}
```

public interface **Set<E>** extends Collection<E>

```
public interface Set<E> extends Collection<E> {
    // Basic operations
    int size();
    boolean isEmpty();
    boolean contains(Object element);
    boolean add(E element);          //optional
    boolean remove(Object element); //optional
    Iterator<E> iterator();

    // Bulk operations
    boolean containsAll(Collection<?> c);
    boolean addAll(Collection<? extends E> c); //optional
    boolean removeAll(Collection<?> c);        //optional
    boolean retainAll(Collection<?> c);        //optional
    void clear();                              //optional

    // Array Operations
    Object[] toArray();
    <T> T[] toArray(T[] a);
}
```

Note: nothing added to Collection interface – except no duplicates allowed

public interface **List<E>** extends Collection<E>

```
public interface List<E> extends Collection<E> {
    // Positional access
    E get(int index);
    E set(int index, E element);    //optional
    boolean add(E element);         //optional
    void add(int index, E element); //optional
    E remove(int index);            //optional
    boolean addAll(int index,
        Collection<? extends E> c); //optional

    // Search
    int indexOf(Object o);
    int lastIndexOf(Object o);

    // Iteration
    ListIterator<E> listIterator();
    ListIterator<E> listIterator(int index);

    // Range-view
    List<E> subList(int from, int to);
}
```

public interface **Queue<E>** extends Collection<E>

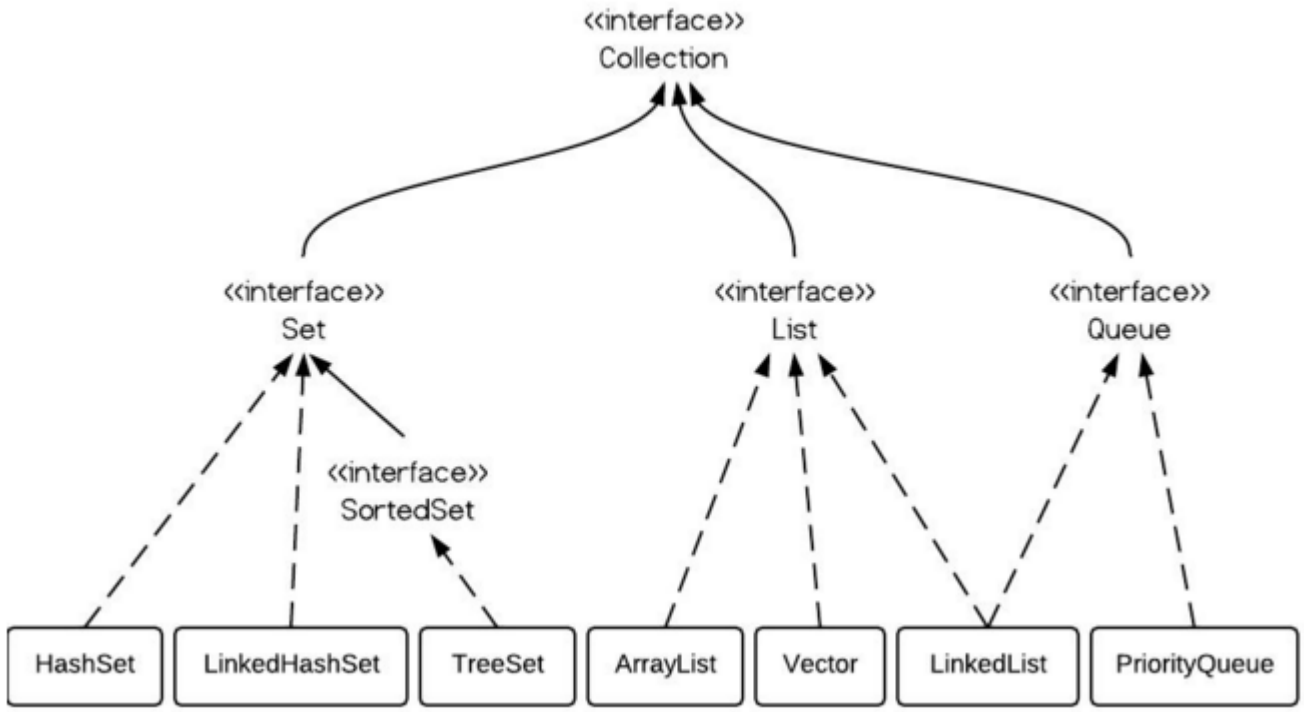
```
public interface Queue<E> extends
 Collection<E> {
    E element();              //throws
    E peek();                 //null
    boolean add(E e);         //throws
    boolean offer(E e);       //add - bool
    E remove();               //throws
    E poll();                 //null
}
```

public interface **Map<K,V>**

```
public interface Map<K,V> {

    // Basic operations
    V put(K key, V value);
    V get(Object key);
    V remove(Object key);
    boolean containsKey(Object key);
    boolean containsValue(Object value);
    int size();
    boolean isEmpty();

    // Bulk operations
    void putAll(Map<? extends K, ? extends V> m);
    void clear();

    // Collection Views
    public Set<K> keySet();
    public Collection<V> values();
    public Set<Map.Entry<K,V>> entrySet();

    // Interface for entrySet elements
    public interface Entry {
        K getKey();
        V getValue();
        V setValue(V value);
    }
}
```

Implementations of Collection

(http://www.programcreek.com/2009/02/the-interface-and-class-hierarchy-for-collections/)



Implementations of Map

(http://www.programcreek.com/2009/02/the-interface-and-class-hierarchy-for-collections/)