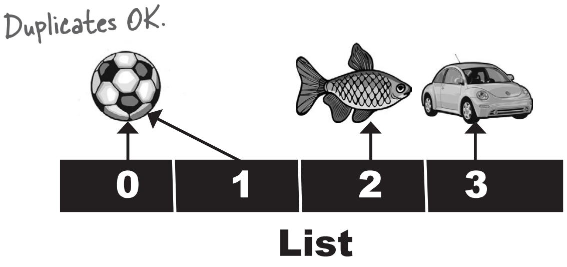
We need a Set instead of a List

From the Collection API, we find three main interfaces, **List**, **Set**, and **Map**. ArrayList is a **List**, but it looks like ***Set*** is exactly what we need.

- **LIST - when *sequence* matters**

  Collections that know about ***index position***.

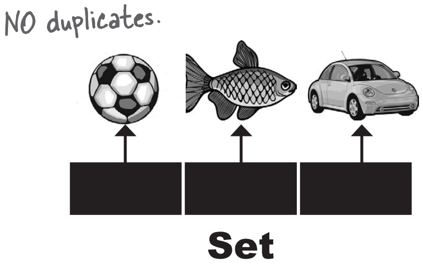
  Lists know where something is in the list. You can have more than one element referencing the same object.



- **SET - when *uniqueness* matters**

  Collections that ***do not allow duplicates***.

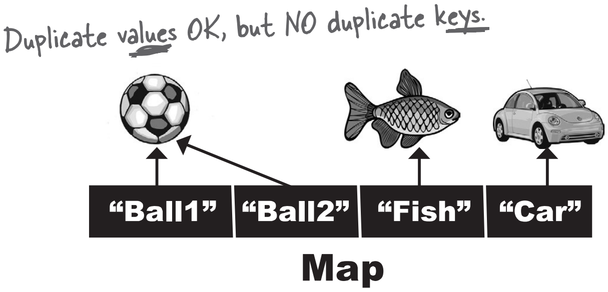
  Sets know whether something is already in the collection. You can never have more than one element referencing the same object (or more than one element referencing two objects that are considered equal—we'll look at what object equality means in a moment).



- **MAP - when *finding something by key* matters**

  Collections that use ***key-value pairs***.

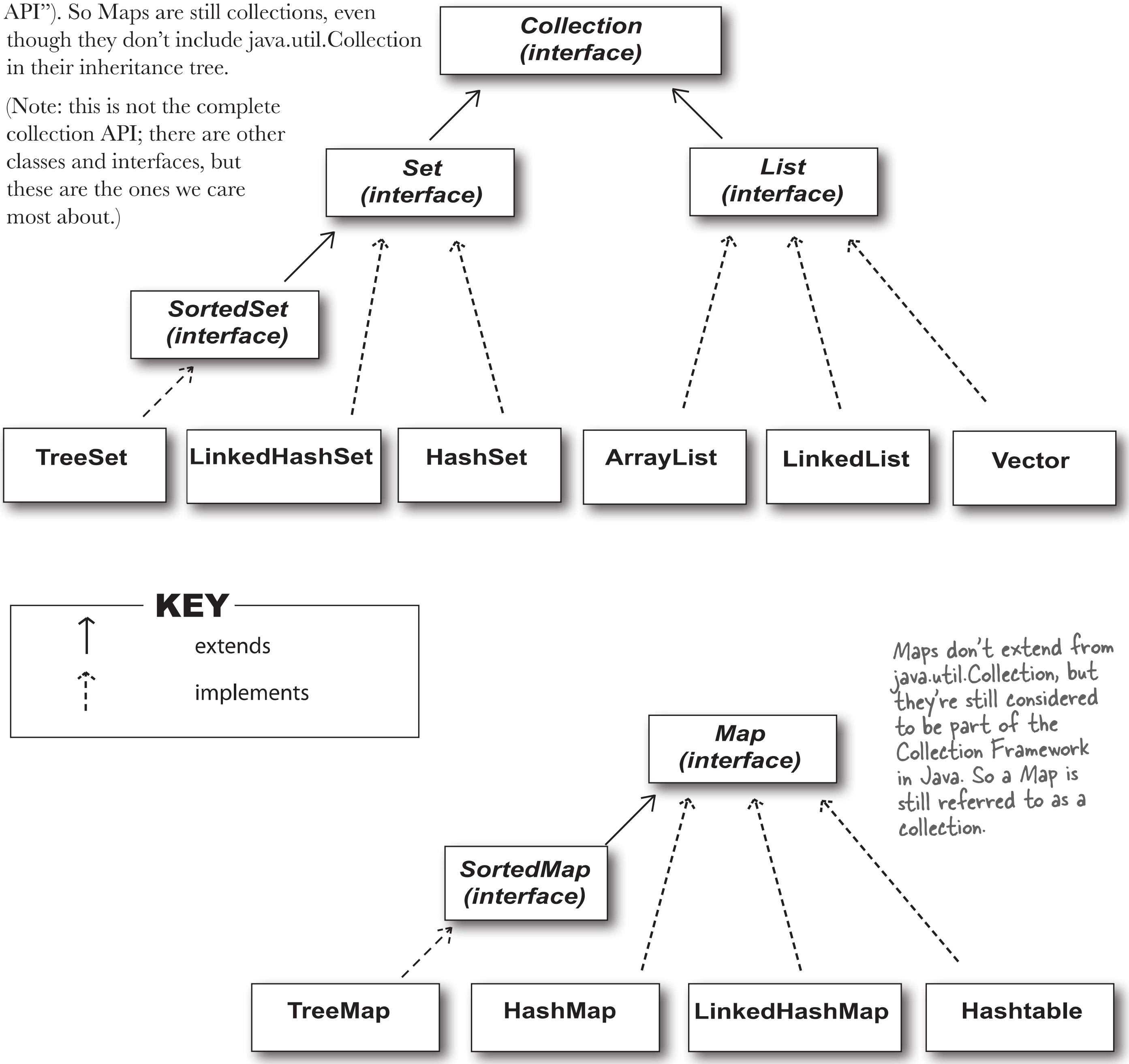
  Maps know the value associated with a given key. You can have two keys that reference the same value, but you cannot have duplicate keys. A key can be any object.



The Collection API (part of it)

Notice that the Map interface doesn't actually extend the Collection interface, but Map is still considered part of the "Collection Framework" (also known as the "Collection API"). So Maps are still collections, even though they don't include java.util.Collection in their inheritance tree.

(Note: this is not the complete collection API; there are other classes and interfaces, but these are the ones we care most about.)



Maps don't extend from java.util.Collection, but they're still considered to be part of the Collection Framework in Java. So a Map is still referred to as a collection.