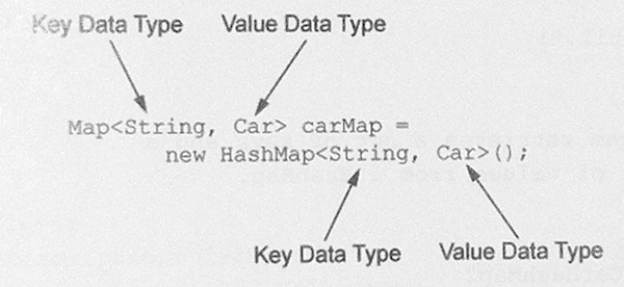
Ch 18.4 Maps

Maps are different. They rely on a pair of values: a key and a value.
Maps are built to efficiently find a value, given a key.
For example: Given a URL (key), find a website's IP address (value). Or, in Java:

```
Map<String, Website> example = new HashMap<String, Website>();
```

Example from page 1099… car name (Ford, Subaru) as key and Car object as value:



For a map `Map<K, V>`: the three most important methods are:

- `put( K key, V value)` - add to the Map… puts the key-value mapping into the Map
- `V get( Object key)` - returns the value associated with the key
- `boolean contains( Object key)` - check the Map… returns true if Map contains a value associated with key

Other important `Map` methods:

- `void clear()` - removes all key-value element from Map
- `boolean isEmpty()` - returns true if Map has no elements
- `Set<K> keySet()` - returns all keys in Map as a Set
- `Collection<V> values()` - returns all values in Map as a Collection

It is *very* common for the key to be a String, and the value to be some more complex "named" structure.
It is common to use those last two methods, `keySet()` and `values()`, translate the Map into another `Collection` data structure.

There are two flavors of `Map`: `HashMap` and `TreeMap`. They use a hash table and binary search tree data structures, respectively. See Note 15 for details on these data structures.