**Java Map WordCount**

Here we'll look at the code for the Wordount problem, which is a classic example of using a Map to process bulk data. (The [Java Map-2](http://codingbat.com/java/Map-2) section has practice problems similar to WordCount.)

Suppose we have an array or strings with lots of duplication, like this...

["a", "b", "a", "f", "b", "a", "z", ....]

We want to figure out which strings appear in the array and how many times each one appears. A map is a great way solve this, and WordCount is a nice example of the technique. Here is the WordCount strategy:

* Create a Map<String, Integer>
* Loop through all the strings
* Use each string as a key into the map
* The Integer value for each key is the number of times that string has been seen
* 2 cases to think about:   
  -The first time we see a string (it is not yet in the map)   
  -Later times we see a string (it is already in the map)

**Try It Yourself**

To try writing this code, here is a problem which is a little easier than wordCount: [word0](http://codingbat.com/prob/p152303)

And here is wordCount itself: [wordCount](http://codingbat.com/prob/p117630)

**wordCount Solution Code**

Here is the WordCount solution code. The key step is distinguishing the first time a string is seen vs. the later times. The code works by storing a count of 1 into the map the first time a string is seen, and then incrementing that count each later time that string is seen.

public Map<String, Integer> wordCount(String[] strings) {

Map<String, Integer> map = new HashMap<String, Integer> ();

for (String s:strings) {

if (!map.containsKey(s)) { // first time we've seen this string

map.put(s, 1);

}

else {

int count = map.get(s);

map.put(s, count + 1);

}

}

return map;

}

The key algorithmic idea here is leveraging the ability of the map to look up a key quickly. When we get to the 10,000th string, we can look up its count quickly, no matter how far back in the array we saw it last.

**More Practice**

Here is a practice problem to do after wordCount: [wordMultiple](http://codingbat.com/prob/p190862)

[CodingBat.com](http://codingbat.com/) code practice.