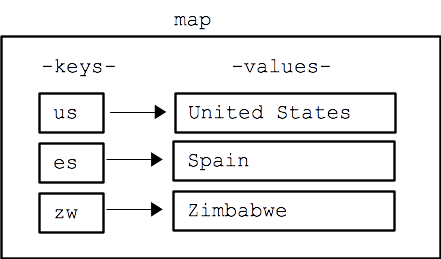
**Java Map Introduction**

Maps are a great data structure for taking in a large amount of data and giving it super-fast access. Here we'll look at the most important Map functions.

A **map** stores key/value pairs, where each **key** has an associated **value**. Given a particular key, the map can look up the associated value very quickly. Here's a map where each key is a 2-letter country-code such as "es", and the associated values are the full country names.



**Map get() method**

The most important Map function is map.get(*key*) which looks up the value for a key and returns it. If the key is not present in the map, get() returns null.

Map<String, String> map = ....(not shown yet)...;

String country;

country = map.get("es"); // returns "Spain"

country = map.get("meh"); // returns null

The get() method returns the value almost instantly, even if the map contains 100 million key/value pairs. Fast performance is the reason maps are great when you have a lot of data to work through.

**Map put() method**

The map.put(*key*, *value*) stores a key/value pair into the map. Any previous value for this key is overwritten. Here is a series of put() calls that builds the above country-code map.

Map<String, String> map = new HashMap<String, String>(); // Make an empty map

map.put("es", "Spain"); // Add the key/value pairs

map.put("us", "United States");

map.put("zw", "Zimbabwe");

The map does not store the key/value pairs in alphabetical order. Instead, the key/value pairs appear in a random order based on the "hashing" strategy used internally. All you need to know is that whey you call get() or put(), the map finds the right key very quickly.

**Map containsKey() method**

The method map.containsKey(*key*) tests if the given key is in the map, returning a boolean.

...

if (map.containsKey("hello")) {

String value = map.get("hello");

// do something with value

The get() method returns null when called with a key not in the map. Therefore, code can check if the value returned by get() is null as another way to check if the key was present. This strategy works fine so long as the code never stores null as a value in the map.

**Map remove() method**

To remove a key/value from a Map, call map.remove(*key*).

**Code Practice**

Practice writing some map code: [mapBully](http://codingbat.com/prob/p197888)

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