# INTRODUCTION TO JAVA

**Java 1.0**

# MAPS

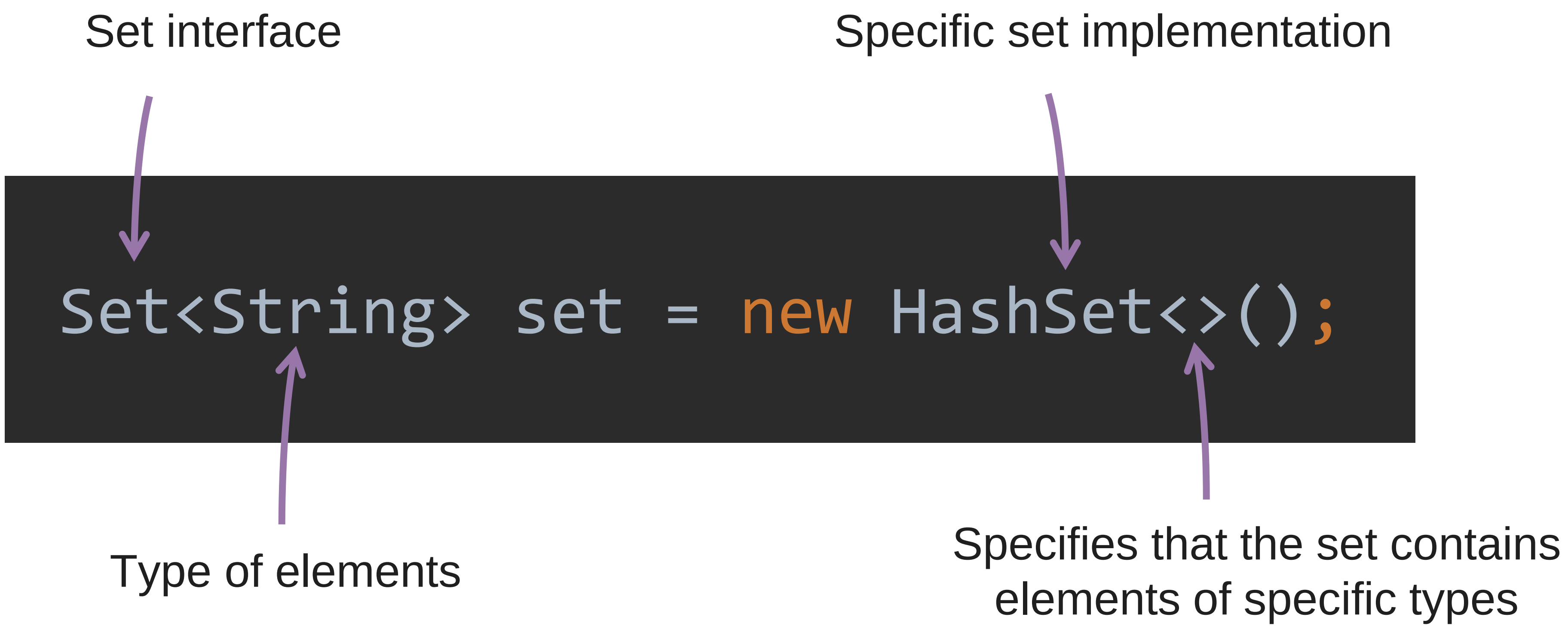
## Lesson # 11

## SET INTERFACE

## SET INTERFACE OVERVIEW

- A collection similar to the List interface
- It doesn’t allow duplicate values
- It’s an unordered collection
- The HashSet is the most commonly used implementation
- The TreeSet is an alternative which is a little more costly

## SET INITIALIZATION



```java
Set<String> set = new HashSet<>();
```

Set interface
Specific set implementation
Type of elements
Specifies that the set contains elements of specific types

{JG} JavaGuru

## BASIC SET OPERATIONS

| Method | Purpose |
|---|---|
| add(Object obj) | Adds a new element at the end of the set |
| remove(Object obj) | Removes the specified element from this set if it is present |
| int size() | Returns the number of elements in this set |
| boolean contains(Object obj) | Returns true if this set contains the specified element |

## ADDING OBJECT TO SET

Code

```
Set<String> countries = new HashSet<>();

countries.add("Latvia");
countries.add("Estonia");
countries.add("Denmark");
countries.add("United States of America");
```

## REMOVING OBJECT FROM SET

Code

```
Set<String> countries = new HashSet<>();
countries.add("Latvia");
countries.add("Estonia");
countries.add("Denmark");
countries.add("United States of America");

countries.remove("Denmark");
```

## CHECKING SET SIZE

Code

```
Set<String> countries = new HashSet<>();
countries.add("Latvia");
countries.add("Estonia");
countries.add("Denmark");

System.out.println("Set size is " + countries.size());
```

Console output

```
Set size is 3
```

## CHECKING SET SIZE - TRICKY

Code

```java
Set<String> countries = new HashSet<>();
countries.add("Latvia");
countries.add("Latvia");
countries.add("Latvia");

System.out.println("Set size is " + countries.size());
```

Console output

```
Set size is 1
```

## CHECKING IF SET CONTAINS ELEMENT

Code

```
Set<String> countries = new HashSet<>();
countries.add("Latvia");
countries.add("Estonia");
countries.add("Denmark");

System.out.println(countries.contains("Latvia"));
System.out.println(countries.contains("United States of America"));
```

Console output

```
true
false
```

## LOOPING THROUGH SET

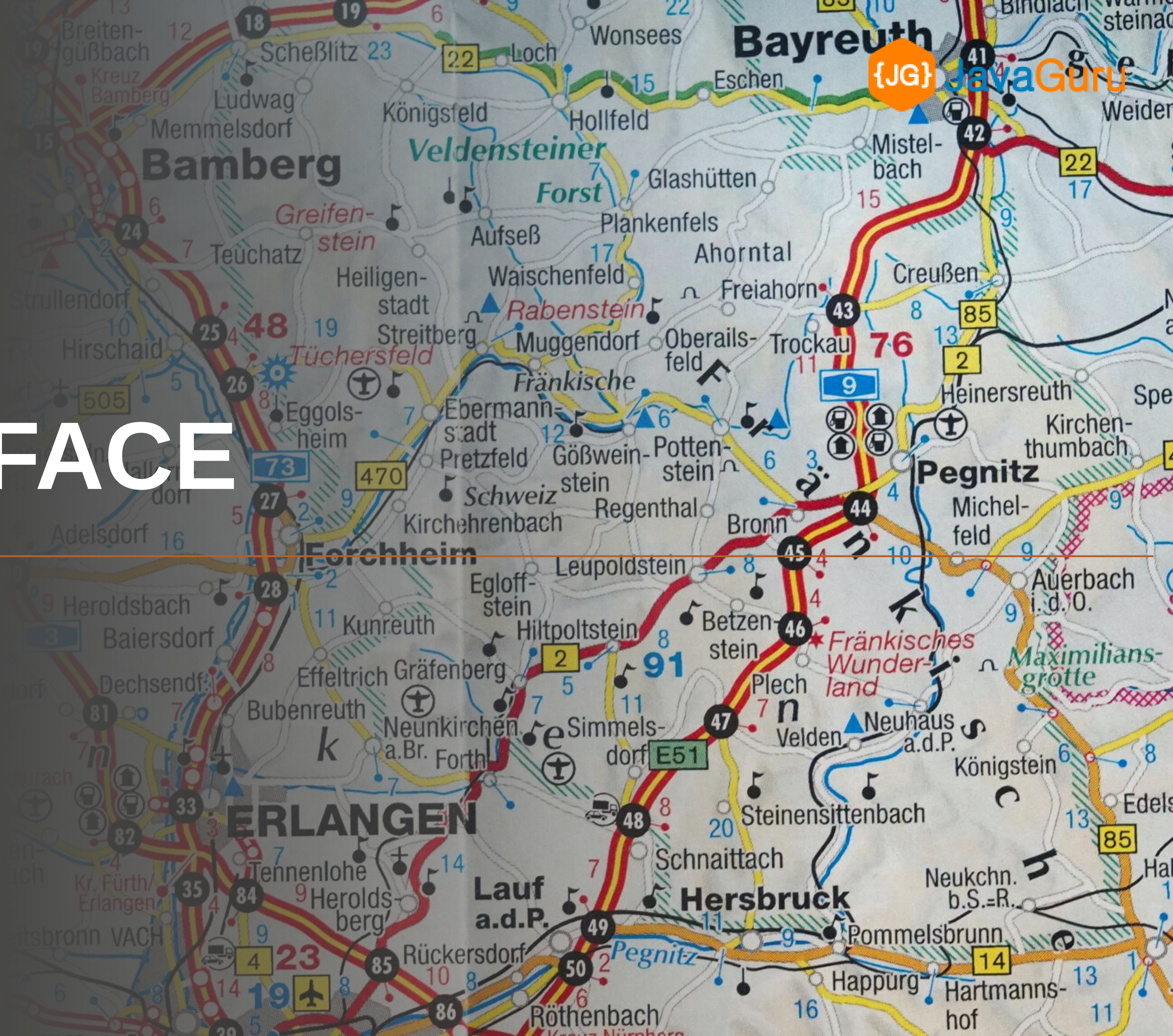
Code

```java
Set<String> countries = new HashSet<>();
countries.add("Latvia");
countries.add("Estonia");
countries.add("Denmark");

for (String country: countries) {
    System.out.println(country);
}
```

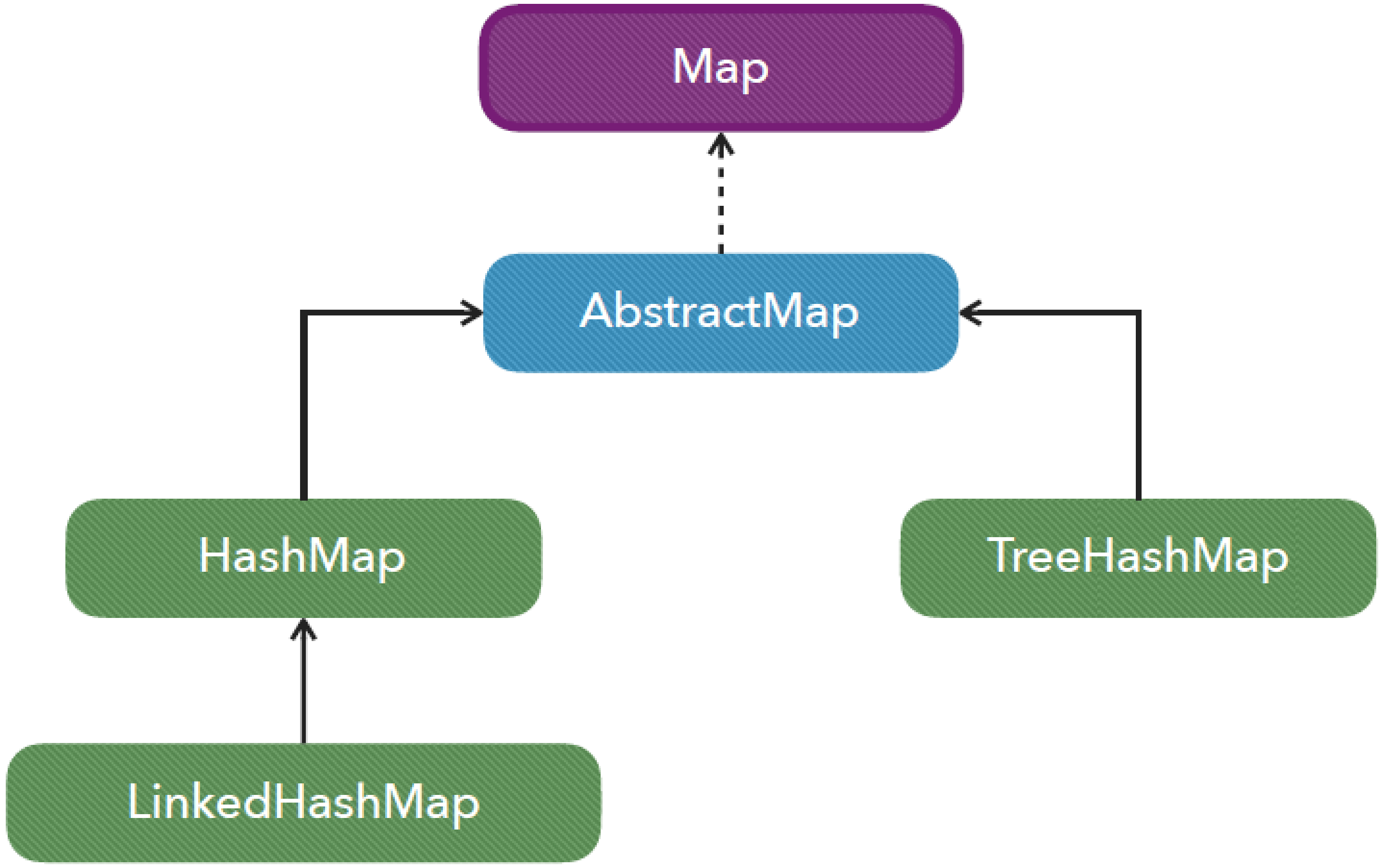
Console output

```
Latvia
Denmark
Estonia
```



# MAP INTERFACE

## MAP API HIERARCHY

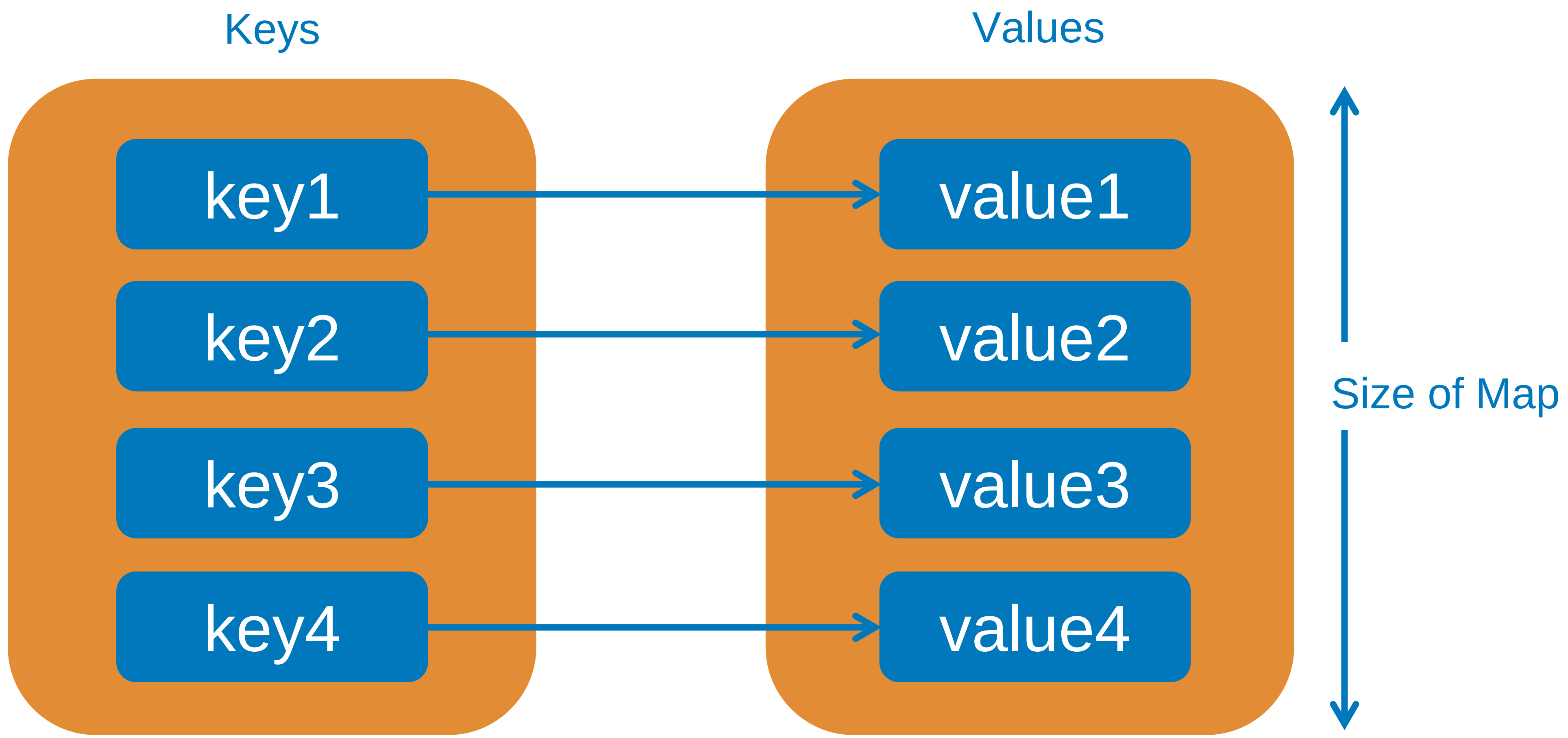


## MAP OVERVIEW

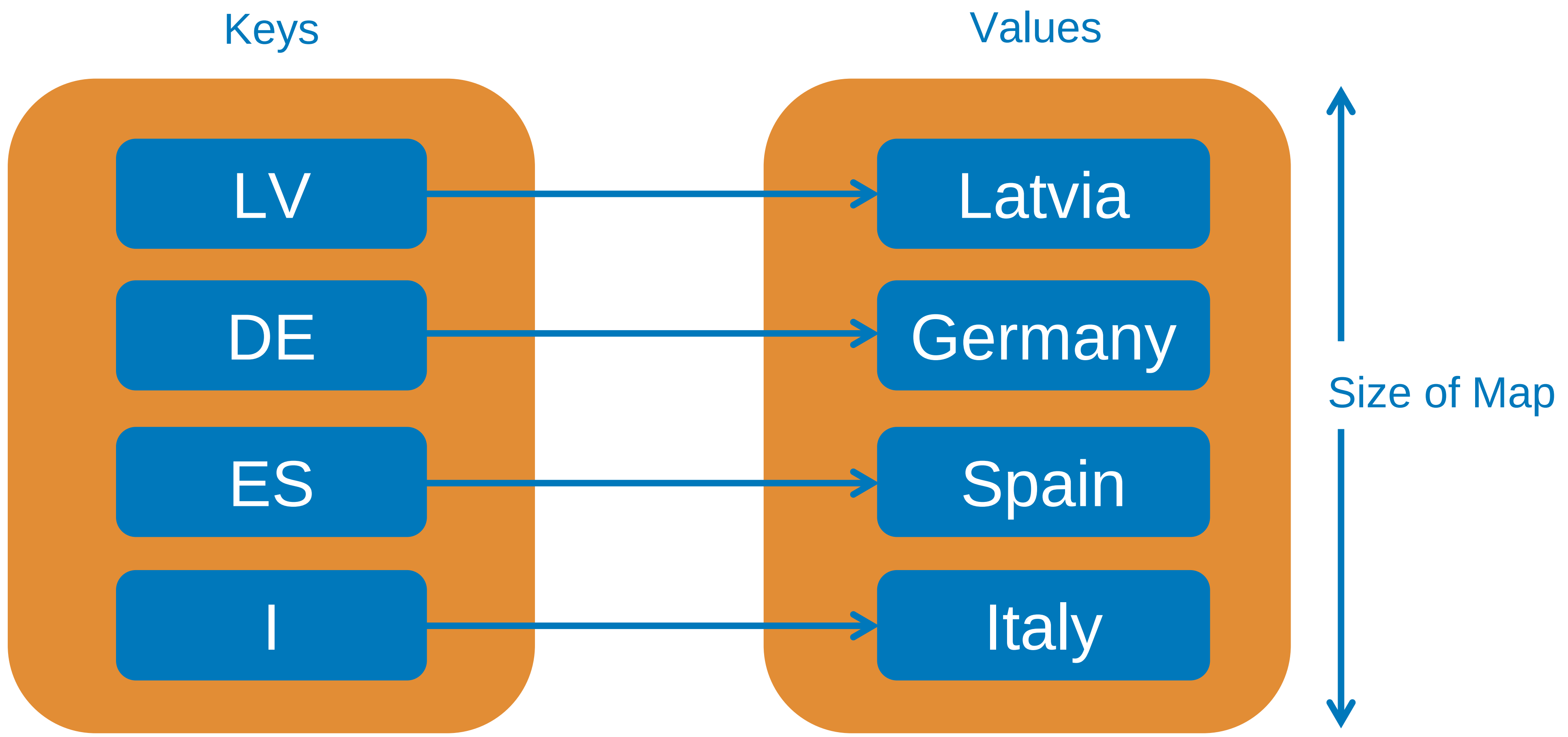
- A Map is a key-value table that can look up any entry by key very efficiently
- "Map" is a general interface of the basic map features, implemented by two main classes:
  - HashMap
  - TreeMap
- A Map stores key-value entries, where each key in the map is associated with a single value.
- It doesn’t allow duplicate keys

{JG} JavaGuru

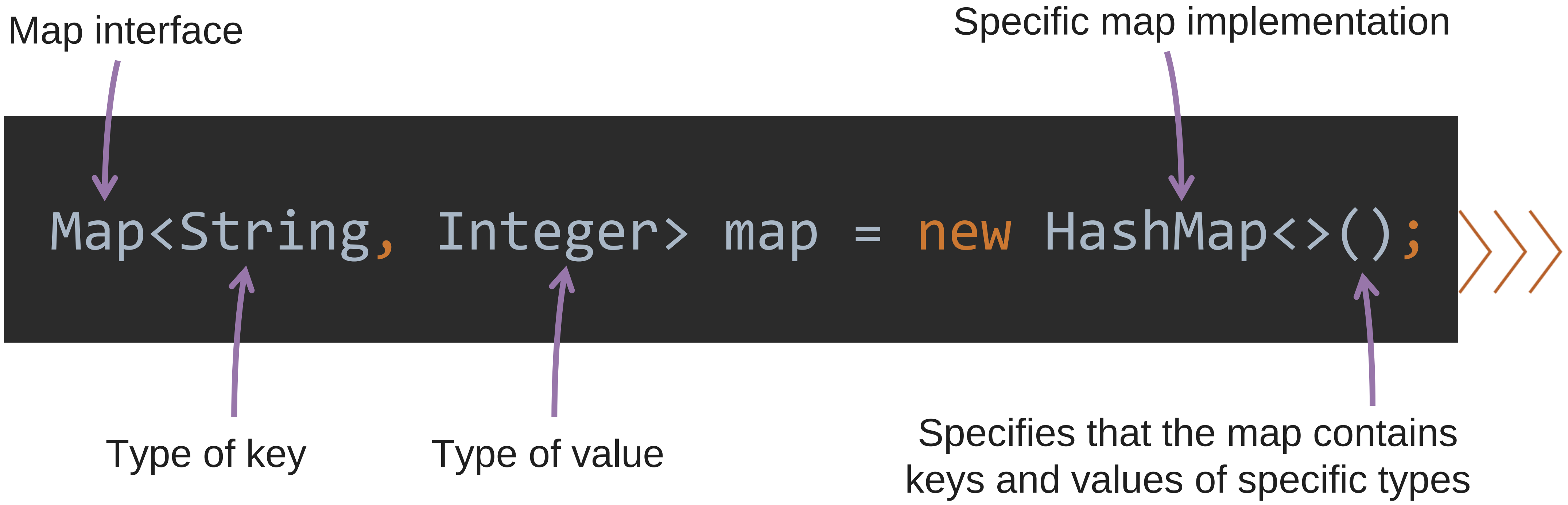
## MAP VISUALISATION



## MAP EXAMPLE



# MAP INITIALIZATION



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

Map interface
Specific map implementation
Type of key
Type of value
Specifies that the map contains keys and values of specific types

## BASIC MAP OPERATIONS

| Method | Purpose |
|---|---|
| put(Object key, Object value) | Puts an entry for the given key into the map with the given value |
| Object get(Object key) | Gets the value previously stored for this key, or null if there is no entry for this key in the map |
| boolean containsKey(Object key) | Returns true if the map contains an entry for the given key |
| int size() | Returns the number of key-value entries in the map |

## PUTTING OBJECT IN MAP

Code

```java
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");
countries.put("I", "Italy");
```

## RETRIEVING OBJECT FROM MAP

Code

```
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");

System.out.println(countries.get("LV"));
System.out.println(countries.get("GB"));
```

Console output

```
Latvia
null
```

{JG} JavaGuru

## CHECKING IF MAP CONTAINS SPECIFIC KEY

Code

```
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");

System.out.println(countries.containsKey("LV"));
System.out.println(countries.containsKey("GB"));
```

Console output

```
true
false
```

## GETTING MAP SIZE

Code

```java
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");
countries.put("GB", "Great Britain");
countries.put("I", "Italy");

System.out.println(countries.size());
```

Console output

```
4
```

## VALUES AND KEYSET

- Collection values:
  - Returns a “live” read-only collection showing all the map values in a random order
  - Iterate over the values collection to see all the values of the map
- Set keyset:
  - Returns a "live" set of all the keys in the map
  - Iterate over the keys set to see all the keys of the map

## LOOPING THROUGH MAP VALUES

Code

```
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");
countries.put("I", "Italy");

for(String countryName : countries.values()) {
    System.out.println(countryName);
}
```

Console output

```
Denmark
Italy
Latvia
```

## LOOPING THROUGH MAP KEYS

Code

```
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");
countries.put("I", "Italy");

for(String countryName : countries.keySet()) {
    System.out.println(countryName);
}
```

Console output

```
DK
I
LV
```

## ENTRY SET

- Values() and keySet() provide the easiest bulk access to a Map;
- The problem with those methods is that they provide access to the keys or the values but not both.
- The entrySet() method provides a Set of special Map.Entry<KEY_TYPE, VALUE_TYPE> objects
- Each Map.Entry object contains one key and one value

## LOOPING THROUGH ENTRY SET

Code

```
Map<String, String> countries = new HashMap<>();
countries.put("LV", "Latvia");
countries.put("DK", "Denmark");
countries.put("I", "Italy");

for(Map.Entry<String, String> entry : countries.entrySet()) {
    System.out.println(entry.getKey() + " - " + entry.getValue());
}
```

Console output

```
DK - Denmark
I - Italy
LV - Latvia
```

# REFERENCES

## REFERENCES

- https://docs.oracle.com/javase/tutorial/collections/interfaces/set.html
- https://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
- https://www.callicoder.com/java-hashset/
- https://www.callicoder.com/java-hashmap/
- https://www.netjstech.com/2015/05/how-hashmap-internally-works-in-java.html
- https://www.netjstech.com/2015/09/how-hashset-works-internally-in-java.html

## QUESTIONS?

{JG} JavaGuru

# THANK YOU!