Maps: Collections of Pairs

Richard Warburton
JAVA CHAMPION, AUTHOR AND PROGRAMMER

@richardwarburto www.monotonic.co.uk

Kvartal *n* (*pl* –er) trimestre *m*; terme *m*
Kvarter *n* (*pl* –er) quart *m* d'heure; quartier *m* (*mil.* et ville); quart *m* d'aune
Kvast *c* (*pl* –e et –er) houppe *f*
kvik *a* vif; éveillé
Kvinde *c* (*pl* –r) femme *f*
†Kvisle *c* (*pl* –r) branche *f* de rivière
Kvist *c* (*pl* –e) 1. petite branche; brindille *f*; 2. mansarde *f*
kvit *a* quitte
kvittere acquitter; donner quittance
Kvittering *c* (*pl* –er) quittance *f*
Kvæg *n* bétail *m*; bestiaux *m*/*pl*
Kvægsølv *n* mercure *m* (= Kviksølv *n*)
kvæle étrangler; étouffer; suffoquer

Kvælstof *n* (gaz) azote *m*; nitrogène *m chem*
kvæste contusionner; **Kvæstning** *c* (*pl* –er) contusion *f*
Kylling *c* (*pl* –er) poulet *m*; poussin *m*
Kyndelmisse *c* la Chandeleur; la Purification (2 févr)
Kyper *c* (*pl* –e) tonnelier *m*; encaveur *m*
Kys *n* (*pl* –) baiser *m*
kysk *a* chaste; **K–hed** *c* chasteté *f*
Kyst *c* (*pl* –er) côte *f*; rivage *m*; bord *m*
Kæde *c* (*pl* –r) chaîne *f* (aussi tissure); collier *m*; suite *f fig*
kæk *a* hardi; audacieux; **K–hed** *c* hardiesse *f*; audace *f*
Kælder *c* (*pl* –e) cave *f*; – **etage** *c* sous-sol *m*; souterrain *m*

Key -> Value

Outline

Why use a map?

Views over maps

Java 8 enhancements

Implementations

Correctly using HashMap

Why Use a Map?

Map API

```
V put(K key, V value)

void putAll(Map<? extends K, ? extends V> values)
```

Adding and Replacing

put for a single value, putAll for another Map

Null keys and values are implementation specific

```
V

get(Object key)


boolean

containsKey(Object key)


boolean

containsValue(Object value)
```

- Looking up elements
- Separate contain methods for key and value
- Objects allow more flexible generic contracts

```
V remove(Object key)

void clear()
```

Removing

Querying Size

```
int size()
```

```
boolean isEmpty()
```

Collection and Map

Map is the only collections that don't extend or implement the Collection interface

Views over Maps

Java 8 Enhancements


EG
EARL GREY
10 TEA BAGS
WISSOTZKY TEA

Altering and Removing

replace(key, value)
Update a single value

replaceAll(BiFunction<K, V, V>)
Replace elements using a function

remove(key, value)
Remove a key only if it has a value

Updating

getOrDefault

putIfAbsent

compute

computeIfAbsent

computeIfPresent

merge

forEach

Convenient callback based iteration

Java 8 Demo

Implementations

General Purpose Implementations

HashMap

Good general purpose implementation

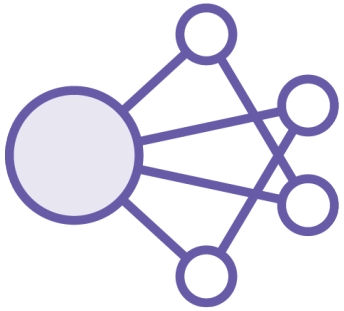
TreeMap

Defines sort order and adds functionality

HashMap

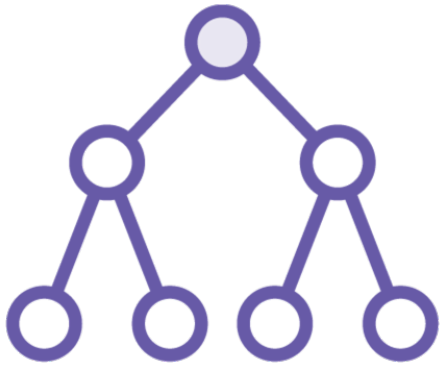
- Good general purpose implementation
- Uses the .hashcode() method
- Maintains an array of buckets
- hash % bucket_count
- Buckets are linked lists to accommodate collisions
- Buckets can be trees
- The number of buckets increases with more elements

TreeMap



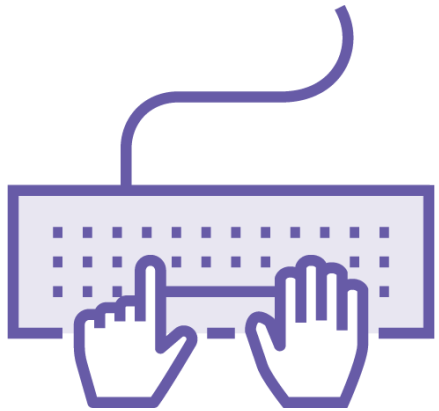
Comparator

Key elements have a sort order



Red / Black Tree

A Balanced binary tree



Navigable & Sorted

Provides functionality that HashMap doesn't

Performance Comparison

	put	get	containsKey	next
HashMap	O(N) , Ω(1)	O(N), Ω(1)	O(N), Ω(1)	O(Capacity/N)
TreeMap	O(log(N))	O(log(N))	O(log(N))	O(log(N))

Correctly Using HashMap

Summary

Maps associate keys and values

2 key implementations

API still improving in Java 8 and beyond

Whatever you need, Java has you covered

Map Visualiser

https://github.com/RichardWarburton/map-visualiser