Collection Operations and Factories

Richard Warburton
JAVA CHAMPION, AUTHOR AND PROGRAMMER

@richardwarburto www.monotonic.co.uk

Overview

Factories

How to make unmodifiable, immutable, empty or wrapping collections

Operations

Useful collection algorithms

Factory Methods (Live Coding)

Factory Method Options (Live Coding)

Factory Methods

```
List<String> list = Collections.emptyList();

Map<Integer, String> map = Collections.emptyMap();

Set<Integer> set = Collections.emptySet();
```

Empty Collections

Immutable

Use when you want to pass a no values to a method that takes a collection

```
List<String> list = Collections.singletonList(“one”);

Map<Integer, String> map = Collections.singletonMap(1, “one”);

Set<Integer> set = Collections.singleton(1);
```

Singletons

Immutable single value of collection

Use when you want to pass a single value to a method that takes a collection

```
List<String> list =
  List.of("UK", "USA");


Map<String, Integer> map =
  Map.of("UK", 67, "USA", 328);


Map<String, Integer> entries =
  Map.ofEntries(
    Map.entry("UK", 67),
    Map.entry("USA", 328));
```

- Collection factories
- Alternative to collection literals
- Runtime immutable – add throws exception
- Overloads for performance
- Alternative map

Immutable Copies

```
// Modifying countries does not modify immutableCountries
Collection<String> countries = new ArrayList();
countries.add("UK“); countries.add("USA");

List<String> immutableCountries = List.copyOf(countries);

Map<String, Integer> populations = new HashMap<>();
populations.put("UK", 67); populations.put("USA", 328);

Map<String, Integer> immutablePopulations = Map.copyOf(populations);
```

Unmodifiable Views

```
// Modifying countries is the only way to modify countriesView
List<String> countries = new ArrayList<>();
countries.add("UK"); countries.add("USA");

List<String> countriesView = Collections.unmodifiableList(countries);

Map<String, Integer> populations = new HashMap<>();
populations.put("UK", 67); populations.put("USA", 328);

Map<String, Integer> populationsView = Collections.unmodifiableMap(populations);
```

Collection Operations

Summary

Collections aren't just about data structures

Common operations ship with the JDK

Immutable + unmodifiable collections reduce scope for bugs