

Java 8 Cheatsheet by Nikolche Mihajlovski, OhmDB

(<http://www.ohmdb.com>)

Lambda expression - easy way to implement single-method (aka functional) interface

Lambda syntax

```
(parameters) -> expression  
(parameters) -> statement  
(parameters) -> { statements }
```

Lambda expression examples

```
(int x, int y) -> x + y  
() -> System.out.println("hi " + s);  
(String s) -> { int n = s.length(); return n; }
```

Run a Runnable

```
Runnable r = () -> System.out.println("Hello!");  
r.run();
```

The PI function

```
Callable<Double> pi = () -> 3.14;  
Double p = pi.call();
```

Sort strings by length

```
String[] words = {"aaa", "b", "cc"};  
Arrays.sort(words, (s1, s2) -> s1.length() - s2.length());  
  
// equivalent to:  
Arrays.sort(words, (String s1, String s2) -> s1.length() - s2.length());
```

Effectively final variables can be referenced in lambdas

```
// s is effectively final (not changed anywhere)  
String s = "foo";  
  
// s can be referenced in the Lambda  
Runnable r = () -> System.out.println(s);
```

Method reference - easy way to use existing method in a functional way

Static method reference

```
// Class::staticMethod syntax
Arrays.sort(items, Util::compareItems);

// equivalent to:
Arrays.sort(items, (a, b) -> Util.compareItems(a, b));
```

Instance method reference

```
// instance::instanceMethod syntax
items.forEach(System.out::print);

// equivalent to:
items.forEach((x) -> System.out.print(x));
```

Reference to a method of arbitrary instance

```
// Class::instanceMethod syntax
items.forEach(Item::publish);

// equivalent to:
items.forEach((x) -> { x.publish(); });
```

Constructor reference

```
ConstructorReference cref = Item::new;
Item item = cref.constructor();
```

Default method - interface method with default implementation

Defining default methods in interfaces

```
interface Descriptive {

    default String desc() {
        return "fantastic";
    }

}
```

Implementing interface with default method

```
class Item implements Descriptive { }
```

```
Item x = new Item();
```

```
// prints "fantastic"
```

```
System.out.println(x.desc());
```

Stream - sequence of values

Count the non-empty strings

```
List<String> strings = ...;
```

```
long n = strings.stream().filter(x -> !x.isEmpty()).count();
```

Join item titles

```
List<Item> items = ...;
```

```
String names = items.stream().map((x) -> x.getTitle()).collect(Collectors.joining(", "));
```

Get distinct countries from cities

```
List<City> cities = ...;
```

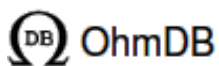
```
List<Country> countries = cities.stream().map((c) -> c.getCountry()).distinct().collect(Collectors.toList());
```

Get count, min, max, sum, and average statistics for items rating

```
List<Item> items = ...;
```

```
IntSummaryStatistics stats = items.stream().mapToInt((x) -> x.getRating()).summaryStatistics();
```

Sponsors:



```
OhmDB db = Ohm.db("my.db");
```

```
Table<Item> items;
```

```
items = db.table(Item.class);
```

```
Item foo = new Item("foo");
```

```
long id = items.insert(foo);
```

[OhmDB - The Irresistible Database for Java \(http://www.ohmdb.com\)](http://www.ohmdb.com)

(<http://empty>)

All articles:

[Caching with ConcurrentHashMap and computeIfAbsent \(caching-with-ConcurrentHashMap-in-java-8.html\)](#)

[Java 8 Cheatsheet \(/\)](#)

[Introduction to Java 8 Lambda expressions \(introduction-to-java-8-lambda-expressions.html\)](#)

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.