Хватит писать циклы! Топ-10 лучших методов для работы с коллекциями из Java 8

[**Статья из группы Random**](https://javarush.ru/groups/general)

1053576 участников

Присоединиться

Что такое коллекции и для чего они нужны, думаю, ученикам JavaRush объяснять не надо. Однако после выхода 8-ой версии многие элементарные операции, на которые раньше уходило 6-7 строчек кода, были упрощены до минимума. Без лишних предисловий — топ-10 лучших методов Java8 Collections Framework, которые сэкономят вам кучу времени и места! Всем привет, друзья! Привычка, как известно, вторая натура. И привыкнув писать for (int i = 0; i <......) переучиваться совсем не хочется (тем более, что конструкция эта довольно проста и понятна). Однако, внутри циклов мы часто повторяем одни и те же элементарные операции, от повторения которых очень хотелось бы избавиться. С выходом Java8 Oracle решили помочь нам с этим. Ниже — 10 лучших методов коллекций, которые сэкономят вам кучу времени и кода

**1. Iterable.forEach(Consumer<? super T> action)** Название говорит само за себя. Перебирает переданную коллекцию, и выполняет лямбда-выражение action для каждого ее элемента.

List <Integer> numbers = **new** ArrayList<> (Arrays.asList(1,2,3,4,5,6));

numbers.forEach(s -> System.out.print(s + " "));

1 2 3 4 5 6

**2. Collection.removeIf(Predicate<? super E> filter)** Тоже ничего сложного. Метод перебирает коллекцию, и удаляет те элементы, которые соответствуют filter.

List <Integer> numbers = **new** ArrayList<>(Arrays.asList(1,2,3,4,5,6,7));

numbers.removeIf(s -> s > 5);

numbers.forEach(s -> System.out.print(s + " "));

В одну строку удаляем из списка все числа больше 5.

**3. Map.forEach(BiConsumer<? super K, ? super V> action)** Метод forEach работает не только для классов, реализующих интерфейс Collection, но и для Map.

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.forEach((a,b) -> System.out.println("Название книги: " + a + ". Автор: " + b));

Название книги: Братья Карамазовы. Автор: Федор Достоевский

Название книги: Философия Java. Автор: Брюс Эккель

Название книги: Преступление и наказание. Автор: Федор Достоевский

Название книги: Война и мир. Автор: Лев Толстой

Название книги: Властелин Колец. Автор: Джон Толкин

**4. Map.compute(K key,BiFunction<? super K, ? super V, ? extends V> remappingFunction)** Выглядит чуть более устрашающе, но на деле прост, как и все предыдущие. Для указанного ключа key этот метод устанавливает в качестве value результат выполнения функции remappingFunction. Например:

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.forEach((a,b) -> System.out.println("Название книги: " + a + ". Автор: " + b));

books.compute("Философия Java", (a,b) -> b+", крутой чувак");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

books.forEach((a,b) -> System.out.println("Название книги: " + a + ". Автор: " + b));

Название книги: Братья Карамазовы. Автор: Федор Достоевский

Название книги: Философия Java. Автор: Брюс Эккель

Название книги: Преступление и наказание. Автор: Федор Достоевский

Название книги: Война и мир. Автор: Лев Толстой

Название книги: Властелин Колец. Автор: Джон Толкин

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название книги: Братья Карамазовы. Автор: Федор Достоевский

Название книги: Философия Java. Автор: Брюс Эккель, крутой чувак

Название книги: Преступление и наказание. Автор: Федор Достоевский

Название книги: Война и мир. Автор: Лев Толстой

Название книги: Властелин Колец. Автор: Джон Толкин

**5. Map.computeIfAbsent(K key, Function<? super K, ? extends V> mappingFunction)** Метод добавит новый элемент в [Map](https://javarush.ru/groups/posts/763-9-glavnihkh-voprosov-o-map-v-java), но только в том случае, если элемент с таким ключом там отсутствует. В качестве value ему будет присвоен результат выполнения функции mappingFunction. Если же элемент с таким ключом уже есть — он не будет перезаписан, а останется на месте. Вернемся к нашим книгам и испробуем новый метод:

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.computeIfAbsent("Гарри Поттер и узник Азкабана", b -> getHarryPotterAuthor());

books.forEach((a,b) -> System.out.println("Название книги: " + a + ". Автор: " + b));

Наша функция mappingFunction:

**public** **static** String getHarryPotterAuthor() {

**return** "Джоан Роулинг";

}

А вот и новая книга:

Название книги: Братья Карамазовы. Автор: Федор Достоевский

Название книги: Философия Java. Автор: Брюс Эккель

Название книги: Преступление и наказание. Автор: Федор Достоевский

Название книги: Война и мир. Автор: Лев Толстой

Название книги: Гарри Поттер и узник Азкабана. Автор: Джоан Роулинг

Название книги: Властелин Колец. Автор: Джон Толкин

**6. Map.computeIfPresent(K key, BiFunction<? super K, ? super V, ? extends V> remappingFunction)** Тот же принцип, что и у Map.compute(), но все вычисления будут выполнены только в случае, если элемент с ключом key уже существует.

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.computeIfPresent("Евгений Онегин", (a,b) -> b="Александр Пушкин");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

books.forEach((a,b) -> System.out.println("Название книги: " + a + ". Автор: " + b));

books.computeIfPresent("Братья Карамазовы", (a,b) -> b="Александр Пушкин");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

books.forEach((a,b) -> System.out.println("Название книги: " + a + ". Автор: " + b));

При первом вызову функции никаких изменений не произошло, потому что книги с названием "Евгений Онегин" в нашей Map нет. А вот во второй раз программа изменила автора для книги "Братья Карамазовы" на "Александр Пушкин". Вывод:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название книги: Братья Карамазовы. Автор: Федор Достоевский

Название книги: Философия Java. Автор: Брюс Эккель

Название книги: Преступление и наказание. Автор: Федор Достоевский

Название книги: Война и мир. Автор: Лев Толстой

Название книги: Властелин Колец. Автор: Джон Толкин

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название книги: Братья Карамазовы. Автор: Александр Пушкин

Название книги: Философия Java. Автор: Брюс Эккель

Название книги: Преступление и наказание. Автор: Федор Достоевский

Название книги: Война и мир. Автор: Лев Толстой

Название книги: Властелин Колец. Автор: Джон Толкин

**7. Map.getOrDefault(Object key, V defaultValue)** Возвращает значение, соответствующее ключу key. Если такой ключ не существует — возвращает значение по умолчанию.

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

String igor = books.getOrDefault("Слово о полку Игореве", "Неизвестный автор");

System.out.println(igor);

Очень удобно:

Неизвестный автор

**8. Map.merge(K key, V value, BiFunction<? super V, ? super V, ? extends V> remappingFunction)** Я даже не стал пытаться посчитать, сколько строк кода вам сэкономит этот метод.

* Если в вашей Map ключ key не существует, или value для этого ключа равно null — метод добавляет в Map переданную пару key-value.
* Если ключ Key существует и его value != null — метод меняет его value на результат выполнения переданной функции remappingFunction.
* Если remappingFunction возвращает null - key удаляется из коллекции.

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.merge("Философия Java", "кто-то там еще", (oldValue, newValue) -> oldValue + " и " + newValue );

books.forEach((a,b) -> System.out.println("Название:" + a + ". Автор: " + b));

Вывод:

Название:Братья Карамазовы. Автор: Федор Достоевский

Название:Философия Java. Автор: Брюс Эккель и кто-то там еще

Название:Преступление и наказание. Автор: Федор Достоевский

Название:Война и мир. Автор: Лев Толстой

Название:Властелин Колец. Автор: Джон Толкин

**9. Map.putIfAbsent(K key, V value)** Раньше чтобы добавить пару в Map, если ее там нет, необходимо было делать следующее:

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

**if** (map.get("Властелин Колец") == null)

map.put("Властелин Колец", "Джон Толкин");

Теперь все стало куда проще:

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

map.putIfAbsent("Властелин Колец", "Джон Толкин");

**10. Map.replace и Map.replaceAll()** Последние по списку, но не по значимости. Map.replace(K key, V newValue — заменяет значение ключа key на newValue, если такой ключ существует. Если нет — ничего не происходит. Map.replace(K key, V oldValue, V newValue) — делает то же самое, но только если текущее значение key равно oldValue. Map.replaceAll(BiFunction<? super K, ? super V, ? extends V> function) — заменяет все значения value на результат выполнения функции function. Например:

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.replace("Братья Карамазовы", "Брюс Эккель", "Джон Толкин");

books.forEach((a,b) -> System.out.println("Название:" + a + ". Автор: " + b));

Название:Братья Карамазовы. Автор: Федор Достоевский

Название:Философия Java. Автор: Брюс Эккель

Название:Преступление и наказание. Автор: Федор Достоевский

Название:Война и мир. Автор: Лев Толстой

Название:Властелин Колец. Автор: Джон Толкин

Не сработало! Текущее значение ключа "Братья Карамазовы" — "Федор Достоевский", а не "Брюс Эккель", поэтому ничего не изменилось.

Map <String, String> books = **new** HashMap<>();

books.put("Война и мир", "Лев Толстой");

books.put("Преступление и наказание", "Федор Достоевский");

books.put("Философия Java", "Брюс Эккель");

books.put("Братья Карамазовы", "Федор Достоевский");

books.put("Властелин Колец", "Джон Толкин");

books.replaceAll((a,b) -> getCoolWriter());

books.forEach((a,b) -> System.out.println("Название:" + a + ". Автор: " + b));

**public** **static** String getCoolWriter() {

**return** "Крутой писатель";

}

Название:Братья Карамазовы. Автор: Крутой писатель

Название:Философия Java. Автор: Крутой писатель

Название:Преступление и наказание. Автор: Крутой писатель

Название:Война и мир. Автор: Крутой писатель

Название:Властелин Колец. Автор: Крутой писатель