Interface Stream<T>

Modifier and Type	Method and Description
boolean	allMatch(Predicate<? super T> predicate) Returns whether all elements of this stream match the provided predicate.
boolean	anyMatch(Predicate<? super T> predicate) Returns whether any elements of this stream match the provided predicate.
OptionalDouble	average() Returns an OptionalDouble describing the arithmetic mean of elements of this stream, or an empty optional if this stream is empty.
<R,A> R	collect(Collector<? super T,A,R> collector) Performs a mutable reduction operation on the elements of this stream using a Collector.
<R> R	collect(Supplier<R> supplier, BiConsumer<R,? super T> accumulator, BiConsumer<R,R> combiner) Performs a mutable reduction operation on the elements of this stream.
long	count() Returns the count of elements in this stream.
Stream<T>	distinct() Returns a stream consisting of the distinct elements (according to Object.equals(Object)) of this stream.
Stream<T>	filter(Predicate<? super T> predicate) Returns a stream consisting of the elements of this stream that match the given predicate.
void	forEach(Consumer<? super T> action) Performs an action for each element of this stream.
<R> Stream<R>	map(Function<? super T,? extends R> mapper) Returns a stream consisting of the results of applying the given function to the elements of this stream.
DoubleStream	mapToDouble(ToDoubleFunction<? super T> mapper) Returns a DoubleStream consisting of the results of applying the given function to the elements of this stream.
IntStream	mapToInt(ToIntFunction<? super T> mapper) Returns an IntStream consisting of the results of applying the given function to the elements of this stream.
LongStream	mapToLong(ToLongFunction<? super T> mapper) Returns a LongStream consisting of the results of applying the given function to the elements of this stream.
Stream<T>	sorted(Comparator<? super T> comparator) Returns a stream consisting of the elements of this stream, sorted according to the provided Comparator.
int	sum() Returns the sum of elements in this stream.
IntSummaryStatistics	summaryStatistics() Returns an IntSummaryStatistics describing various summary data about the elements of this stream.

Class Collectors

Modifier and Type	Method and Description
`static <T,A,R,RR> Collector<T,A,RR>`	`collectingAndThen(Collector<T,A,R> downstream, Function<R,RR> finisher)` Adapts a `Collector` to perform an additional finishing transformation.
`static <T,K> Collector<T,?,Map<K,List<T>>>`	`groupingBy(Function<? super T,? extends K> classifier)` Returns a `Collector` implementing a "group by" operation on input elements of type `T`, grouping elements according to a classification function, and returning the results in a `Map`.
`static <T,K,A,D> Collector<T,?,Map<K,D>>`	`groupingBy(Function<? super T,? extends K> classifier, Collector<? super T,A,D> downstream)` Returns a `Collector` implementing a cascaded "group by" operation on input elements of type `T`, grouping elements according to a classification function, and then performing a reduction operation on the values associated with a given key using the specified downstream `Collector`.
`static <T> Collector<T,?,Optional<T>>`	`maxBy(Comparator<? super T> comparator)` Returns a `Collector` that produces the maximal element according to a given `Comparator`, described as an `Optional<T>`.
`static <T> Collector<T,?,Optional<T>>`	`minBy(Comparator<? super T> comparator)` Returns a `Collector` that produces the minimal element according to a given `Comparator`, described as an `Optional<T>`.
`static<T> Collector<T,?,Map<Boolean,List<T>>>`	`partitioningBy(Predicate<? super T> predicate)` Returns a `Collector` which partitions the input elements according to a `Predicate`, and organizes them into a `Map<Boolean, List<T>>`.
`static <T,D,A> Collector<T,?,Map<Boolean,D>>`	`partitioningBy(Predicate<? super T> predicate, Collector<? super T,A,D> downstream)` Returns a `Collector` which partitions the input elements according to a `Predicate`, reduces the values in each partition according to another `Collector`, and organizes them into a `Map<Boolean, D>` whose values are the result of the downstream reduction.
`static <T> Collector<T,?,List<T>>`	`toList()` Returns a `Collector` that accumulates the input elements into a new `List`.

Interface Comparator<T>

Modifier and Type	Method and Description
int	compare(T o1, T o2) Compares its two arguments for order.
static <T,U extends Comparable<? super U>> Comparator<T>	comparing(Function<? super T,? extends U> keyExtractor) Accepts a function that extracts a Comparable sort key from a type T, and returns a Comparator<T> that compares by that sort key.
static <T,U> Comparator<T>	comparing(Function<? super T,? extends U> keyExtractor, Comparator<? super U> keyComparator) Accepts a function that extracts a sort key from a type T, and returns a Comparator<T> that compares by that sort key using the specified Comparator.
default Comparator<T>	reversed() Returns a comparator that imposes the reverse ordering of this comparator.
default Comparator<T>	thenComparing(Comparator<? super T> other) Returns a lexicographic-order comparator with another comparator.

Class Optional<T>

Modifier and Type	Method and Description
T	get() If a value is present in this Optional, returns the value, otherwise throws NoSuchElementException.
boolean	isPresent() Return true if there is a value present, otherwise false.
T	orElse(T other) Return the value if present, otherwise return other.