The Most Common Intermediate Operations

Up until now, I've kind of glossed over intermediate operations.

I've used filter, limit, map and sorted in my examples.

As you can see from this table, the operations you've already seen briefly, cover half of the basic operations available to your stream pipelines.

Return Type	Operation
`Stream<T>`	`distinct()`

`Stream<T>`	`filter(Predicate<? super T> predicate)`
	`takeWhile(Predicate<? super T> predicate)`
	`dropWhile(Predicate<? super T> predicate)`

`Stream<T>`	`limit(long maxSize)`
`Stream<R>`	`map(Function<? super T,? extends R> mapper)`
`Stream<T>`	`peek(Consumer<? super T> action)`
`Stream<T>`	`skip(long n)`
`Stream<T>`	`sorted()` `sorted(Comparator<? super T> comparator)`

Intermediate Operations that effect the size of the Resulting Stream

I'll start by talking about the set of operations, that may change the number of elements, in the resulting stream.

Return Type	Operation	Description
`Stream<T>`	`distinct()`	Removes duplicate values from the Stream.

`Stream<T>`	`filter(Predicate<? super T> predicate)`
	`takeWhile(Predicate<? super T> predicate)`
	`dropWhile(Predicate<? super T> predicate)`
		These methods allow you to reduce the elements in the output stream. Elements that match the filter's Predicate are kept in the outgoing stream, for the filter and takeWhile operations. Elements will be dropped until or while the dropWhile's predicate is not true.

`Stream<T>`	`limit(long maxSize)`	This reduces your stream to the size specified in the argument.
`Stream<T>`	`skip(long n)`	This method skips elements, meaning they won't be part of the resulting stream.

Declarative Language of Stream Operations Resembles Query commands

The Java API designers designed the Stream to let you process data in a declarative way, much like a structured query language or SQL in a database.

Again this lets you say **what should happen**, and **not actually how it will happen**.

If you've had experience querying databases, you might be familiar with the limit and distinct keywords, available in some database query languages.

The filter operation represents your where clause, and sorted would be your order by clause, and so on.

There are aggregate functions commonly used in queries as well, such as max, min, count and so on.