##### **Stream API**

Stream is a sequence of elements supporting sequential and parallel aggregate operations.

To perform a computation, stream operations are composed into a stream pipeline. A stream pipeline consists of a ****source**** (which might be an array, a collection, a generator function, an I/O channel, etc), zero or more ****intermediate operations**** (which transform a stream into another stream, such as filter(Predicate)), and ****a**** ****terminal operation****(which produces a result or side-effect, such as count() or forEach(Consumer)). ****Streams are lazy; computation on the source data is only performed when the terminal operation is initiated****, and source elements are consumed only as needed.

To create a stream, just call stream() method on any Collection.

For example:

List<String> aList = getAListFromSomePlaces();

aList.stream(); // generate a stream

aList.stream().filter(s->s.startWith("A")); // pipeline the stream, filter out, keep the strings starting with A

aList.stream().filter(s->s.startWith("A")).forEach(System.out::println); // a complete pipeline includes intermediate operation and only 1 termination operation.