lazy Streams

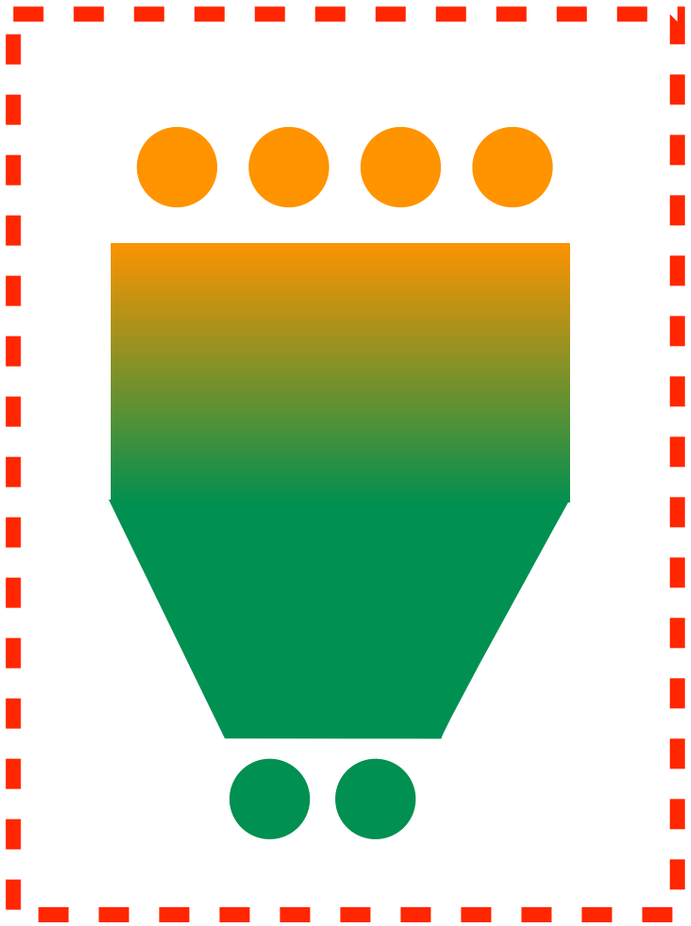
parallel Stream

.methods()

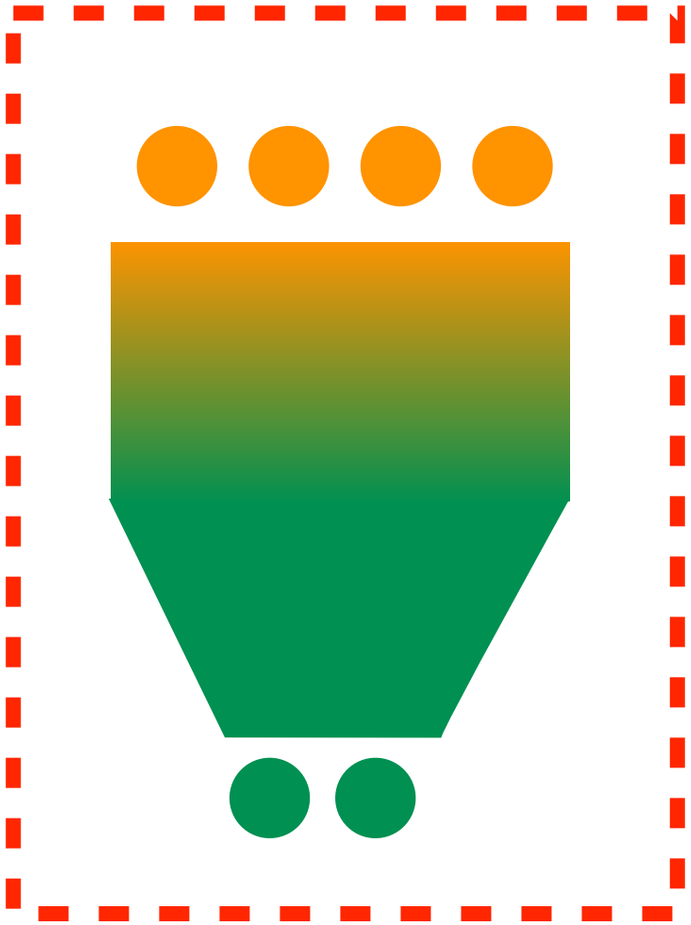
Stream.map( )

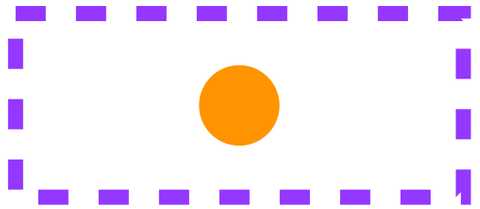
.filter( )

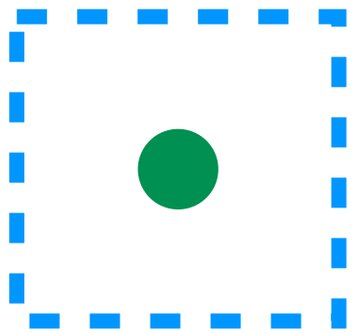
Context



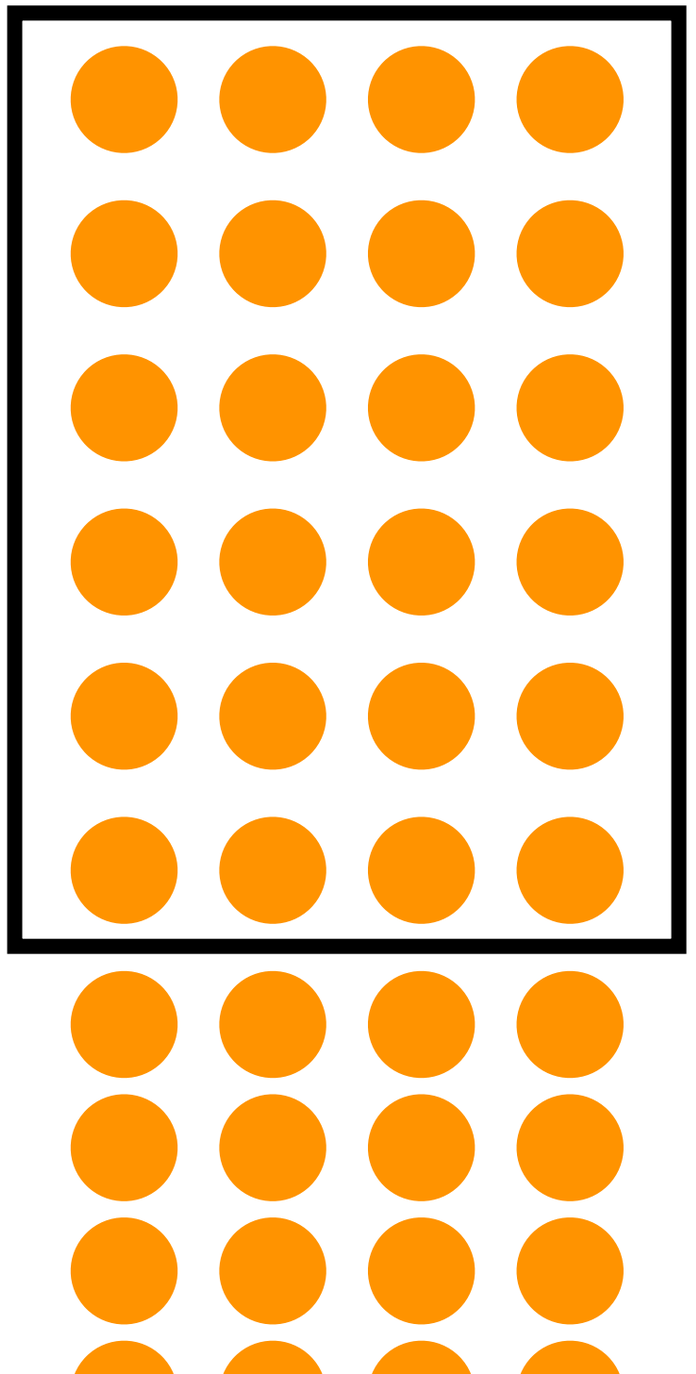
Contexts



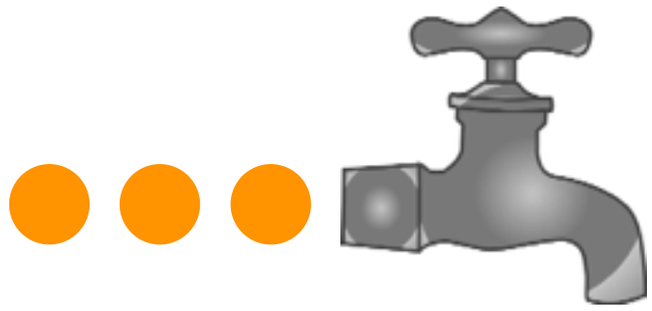





BONUS!



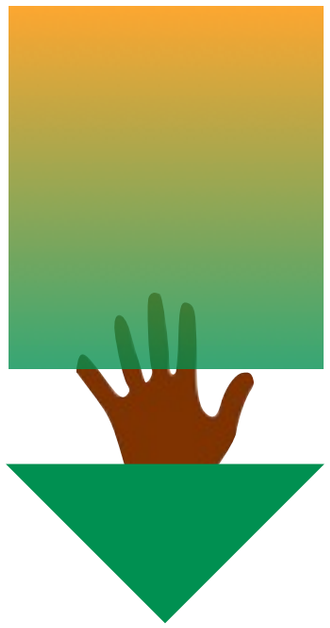

How can Streams be
more efficient than loops?

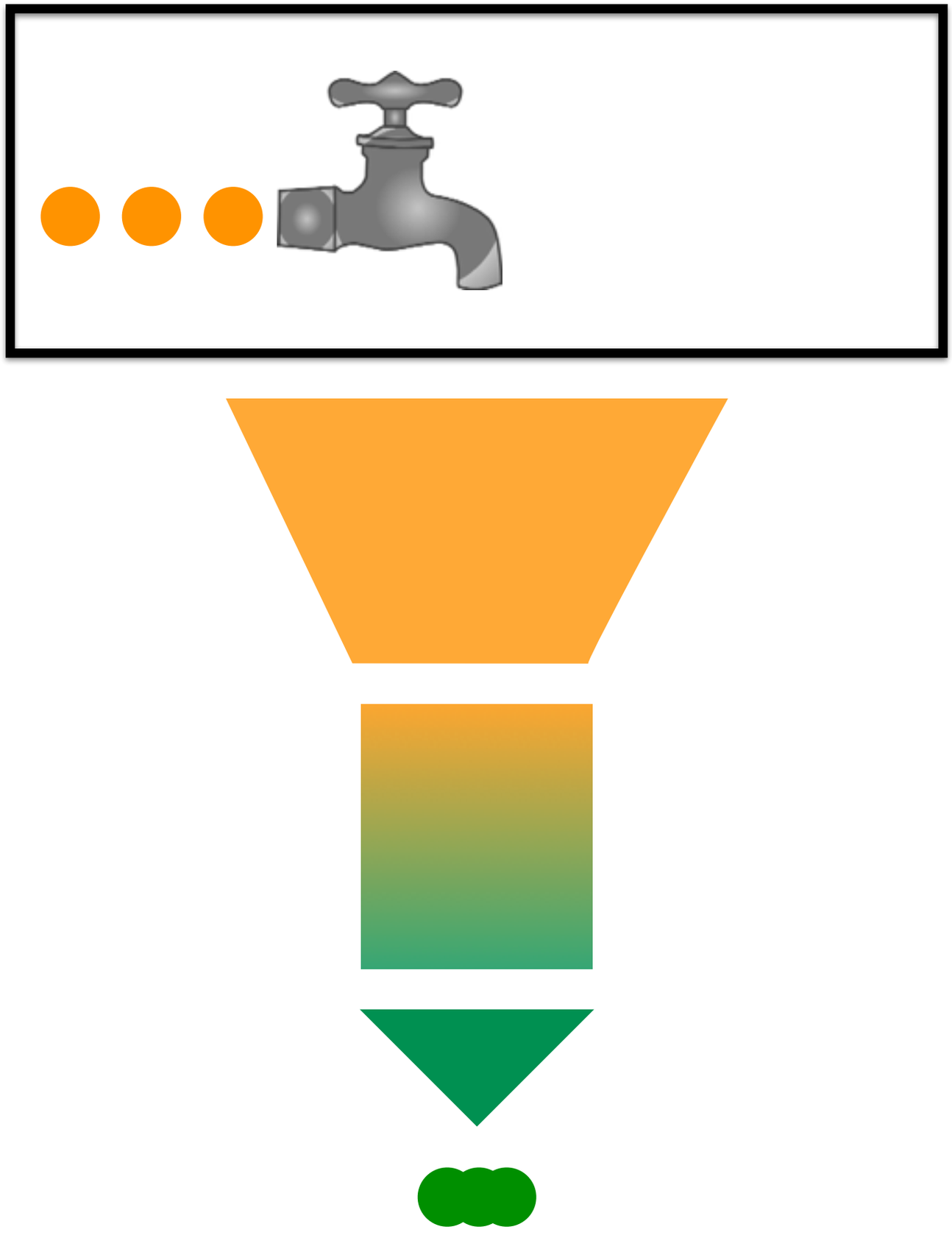


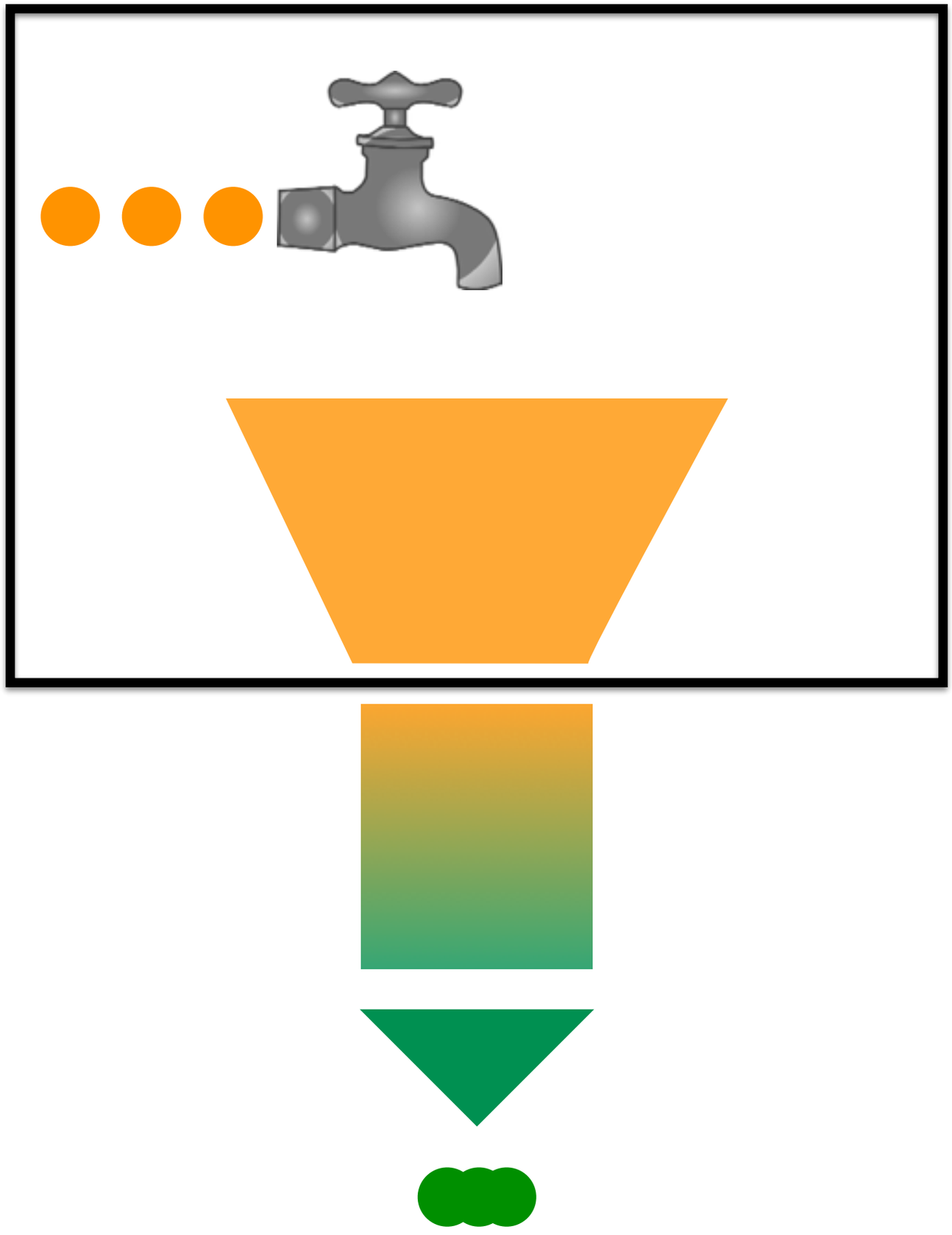
filter

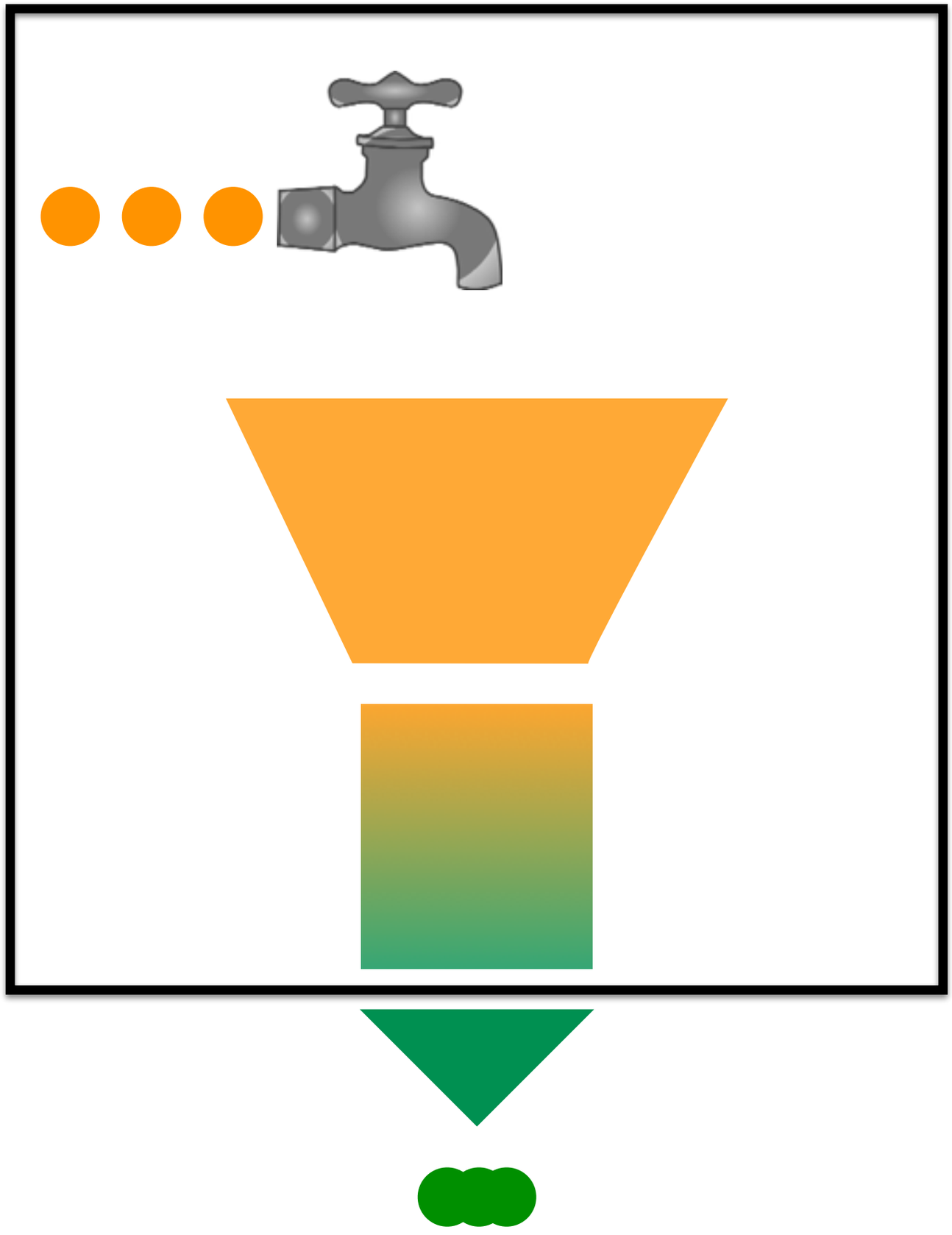
map

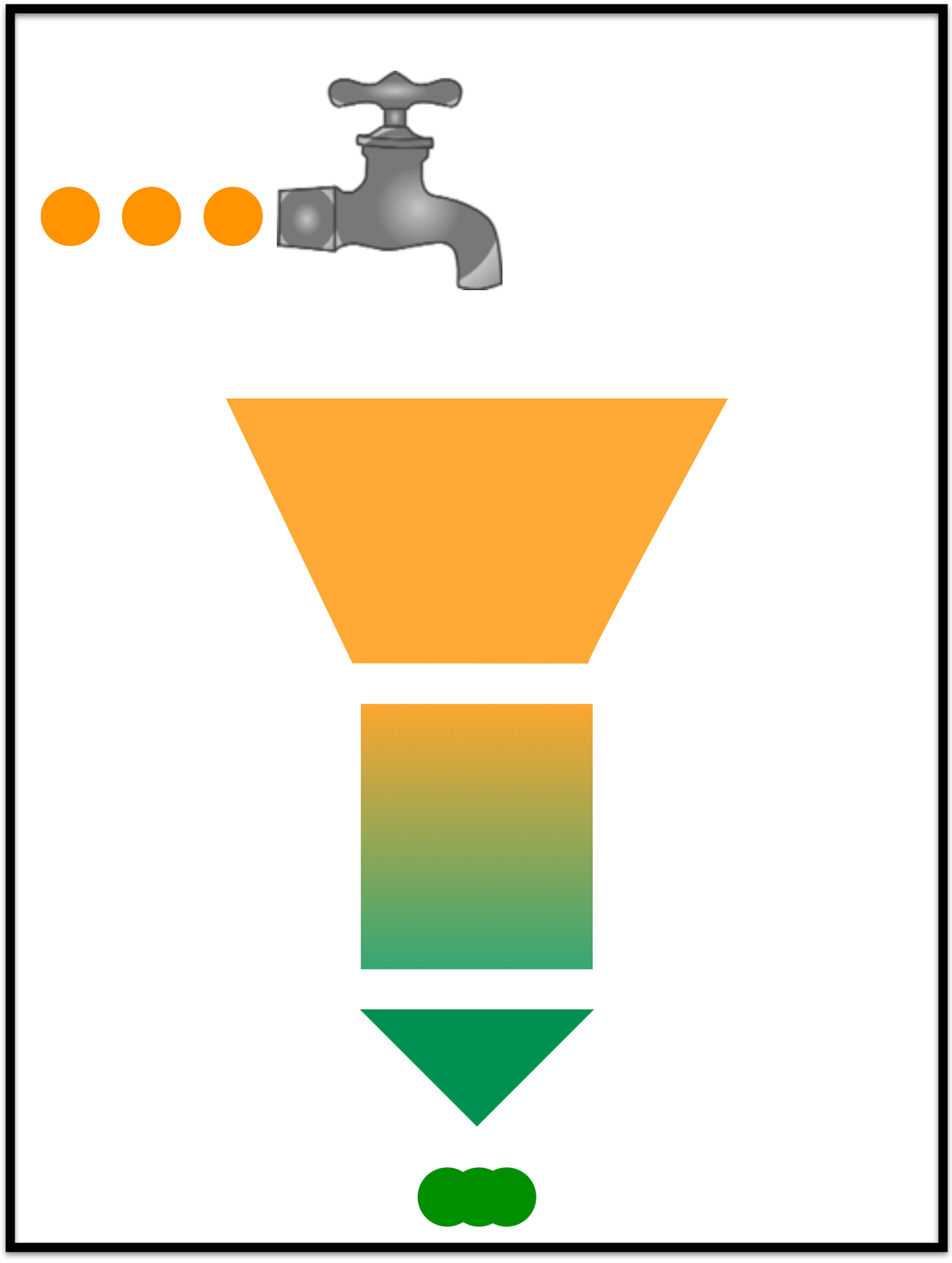
reduce

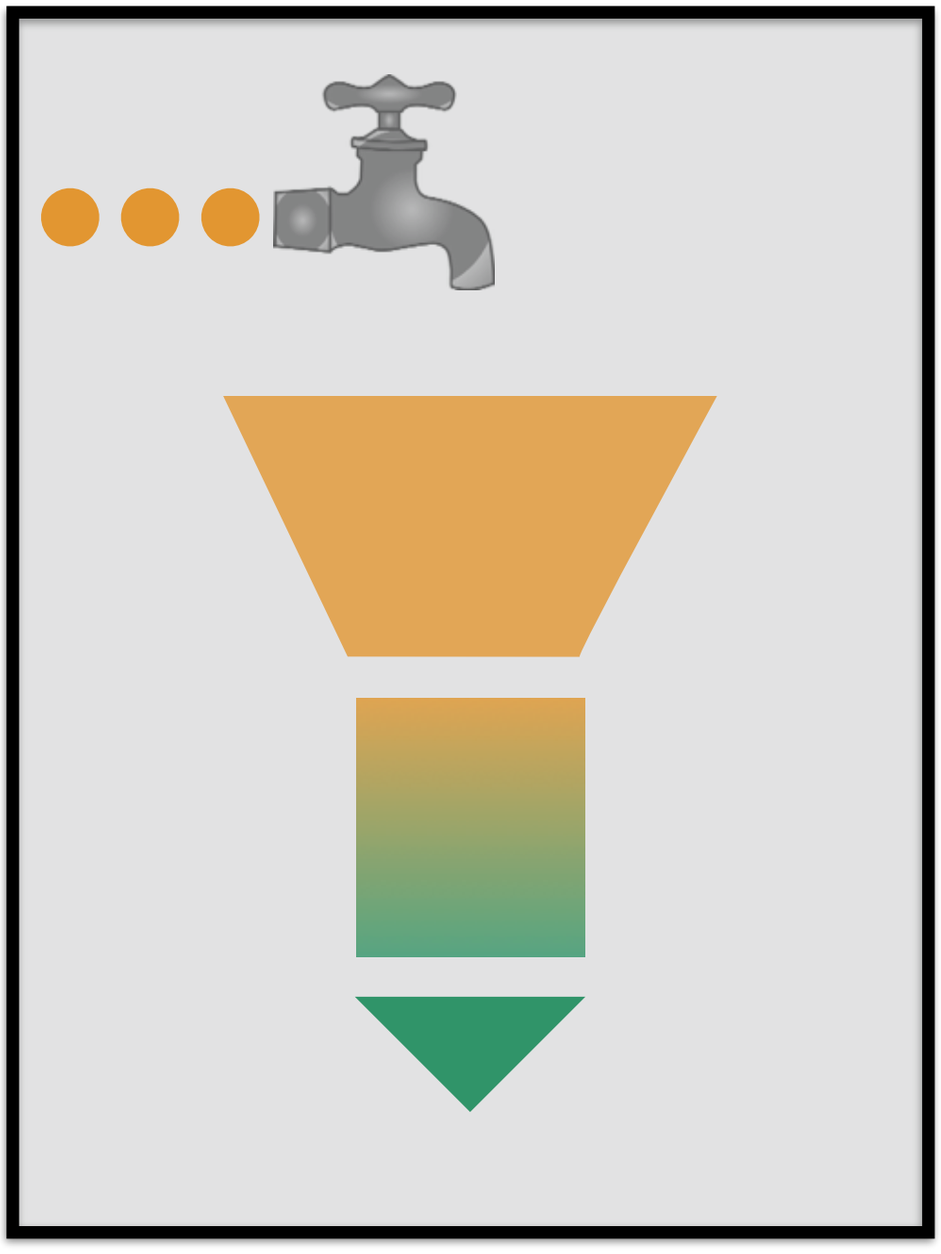


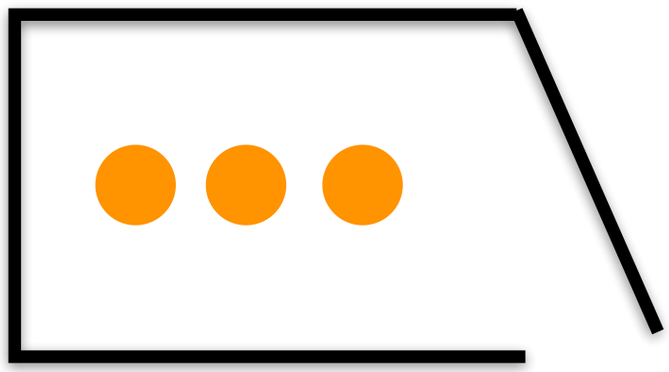


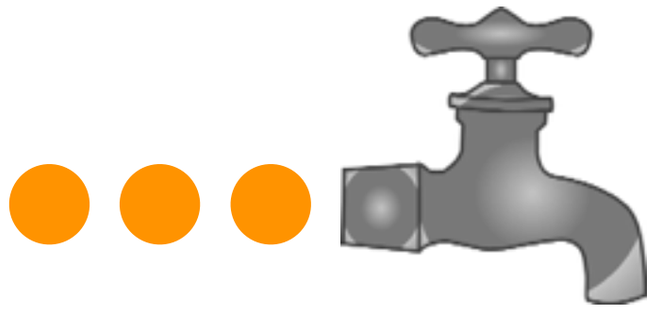




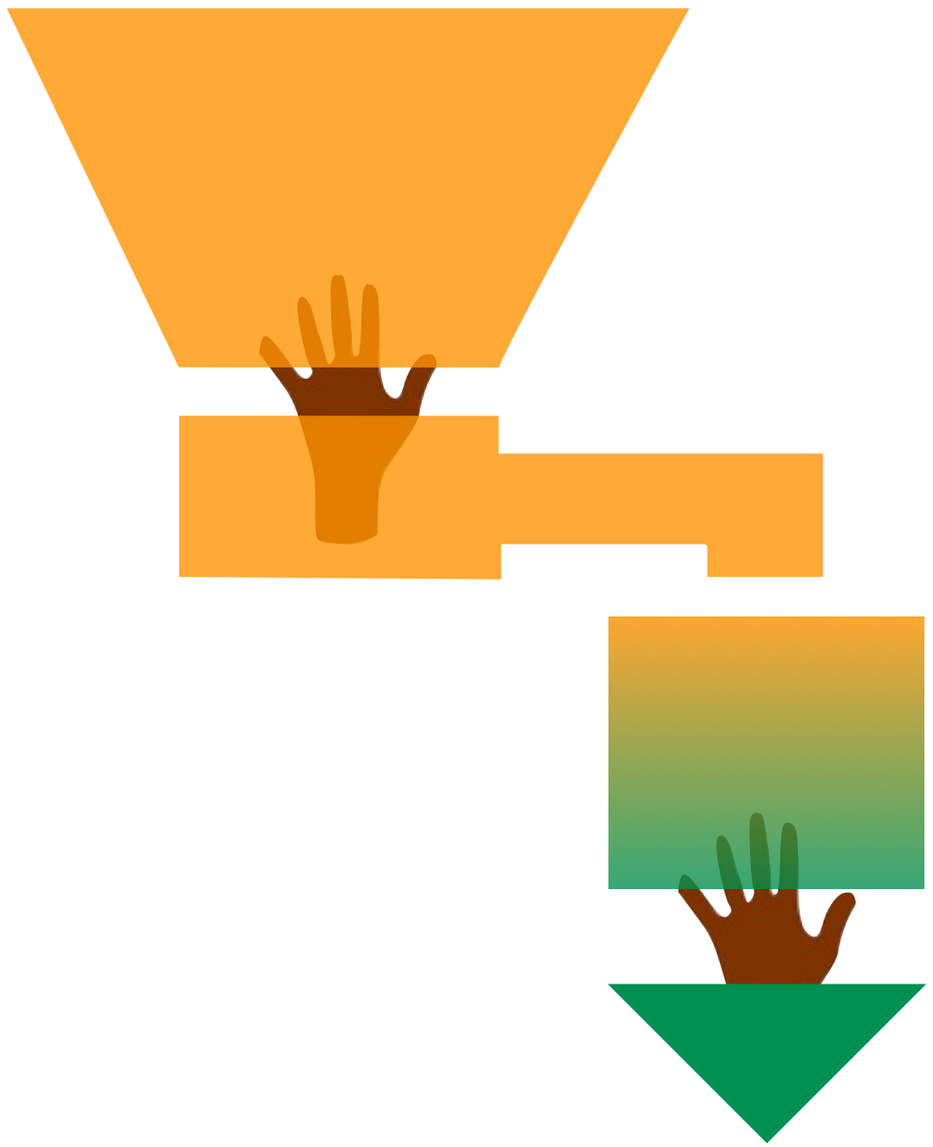








filter



limit

map

reduce

Efficiency

Separation of Concerns

Efficiency

Separation of Concerns

What to do?

When to stop?

Let's see some
Stream operations!

Terminal Operators

trigger processing

return a value

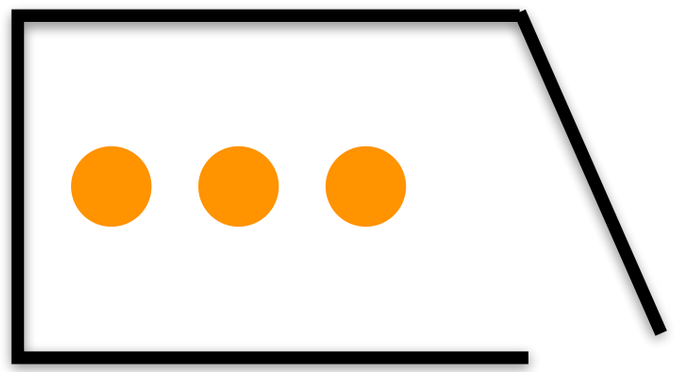
terminate the Stream instance

Intermediate Operators

trigger no processing

return a Stream

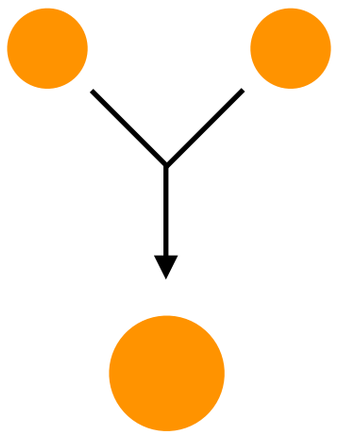
let you keep going



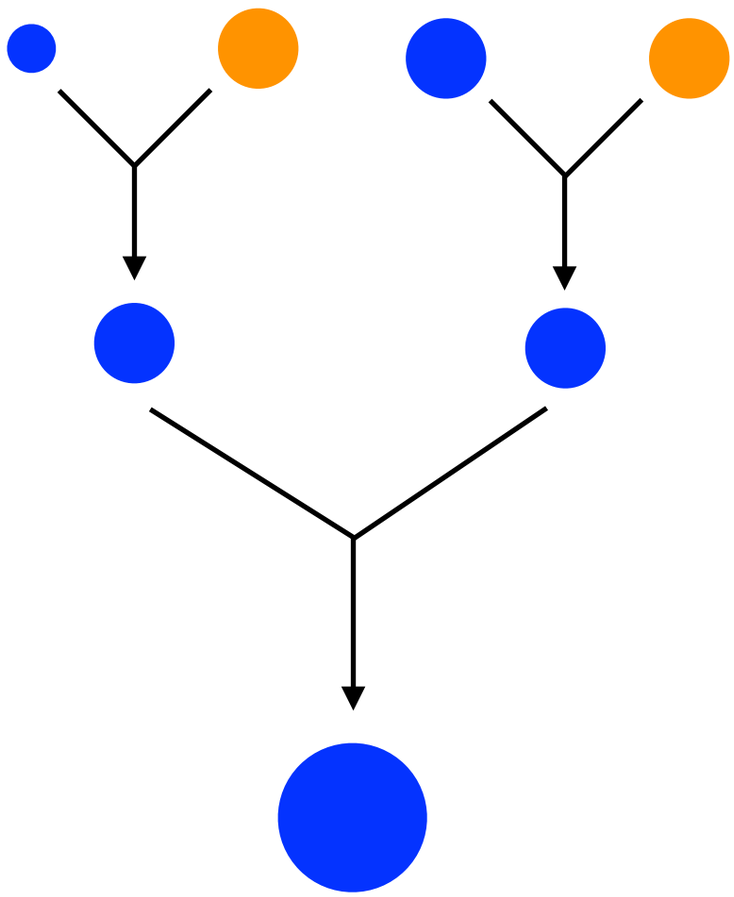
Stream **Iterator**

one-time traversal

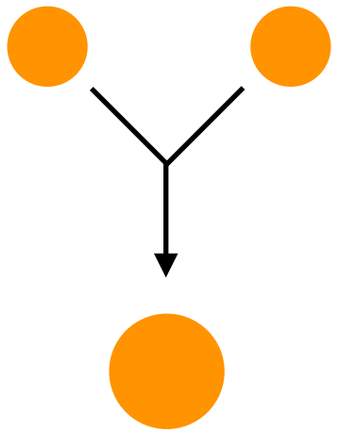
reduce



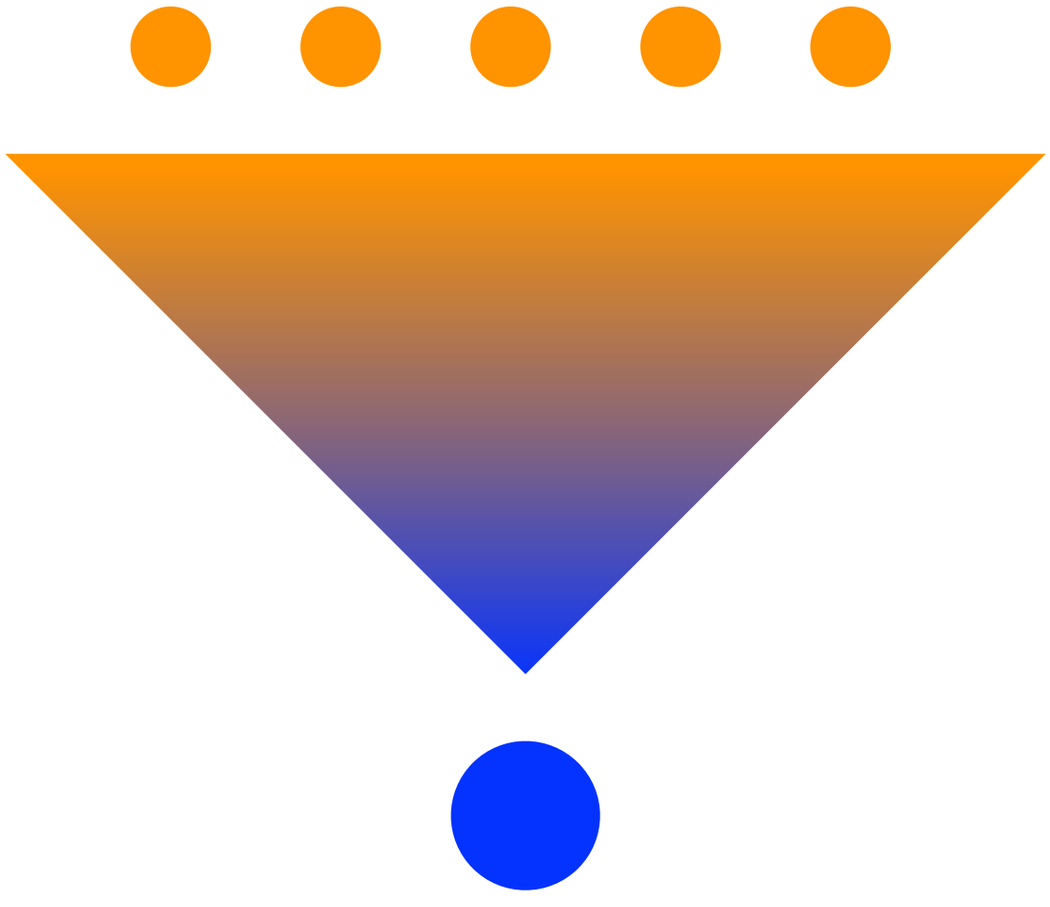
collect



reduce



collect



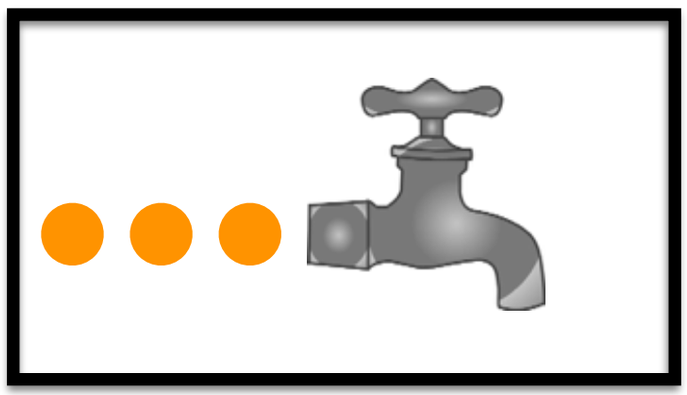
Next up:
parallel Streams!

BUT FIRST…

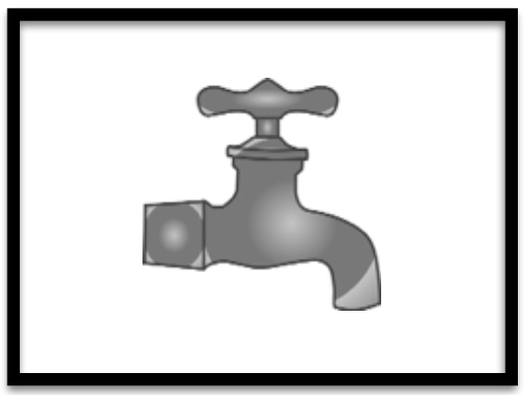
Let’s see some different Sales.

Infinite streams need
short-circuiting operations.

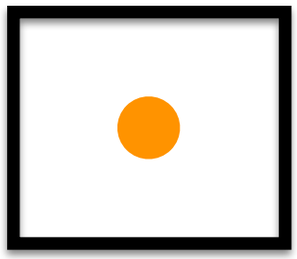
Stream



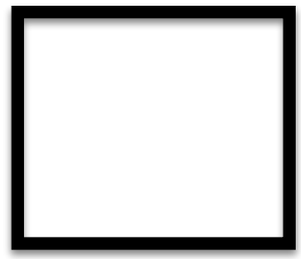
to

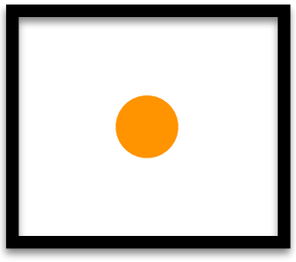


Optional



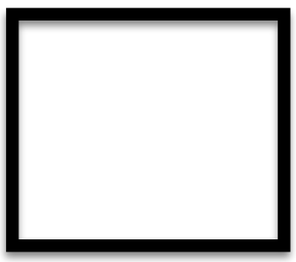
or





isPresent()

or



isEmpty()


OW!
null

Stream

Optional

Function ?

withTransaction( )

start

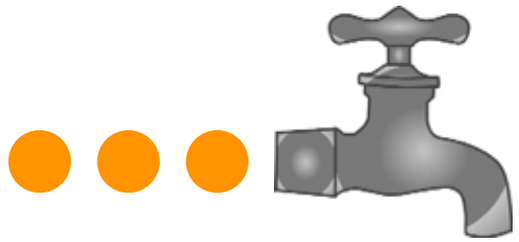
exception handling

end

Review Stream

Java 6

Secret Functional Programmer Terms

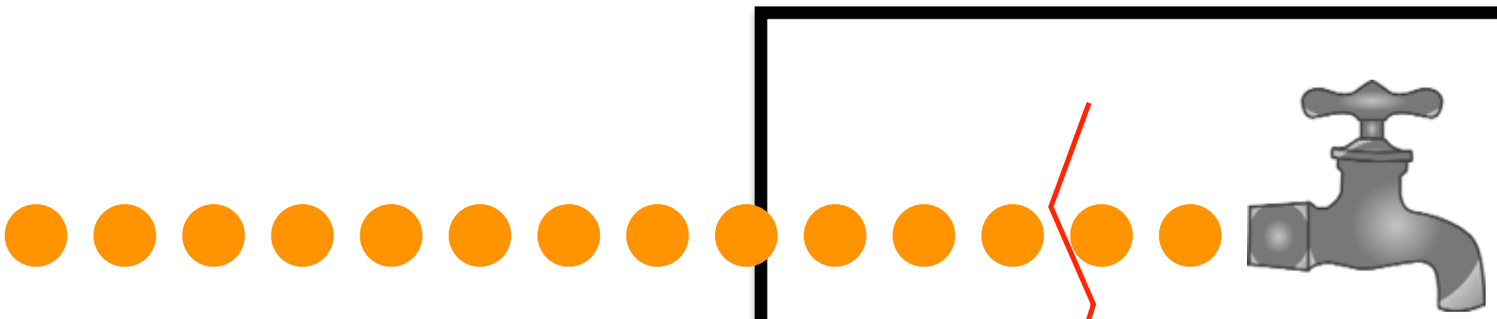


intermediate operation

intermediate operation

intermediate operation

terminal operation

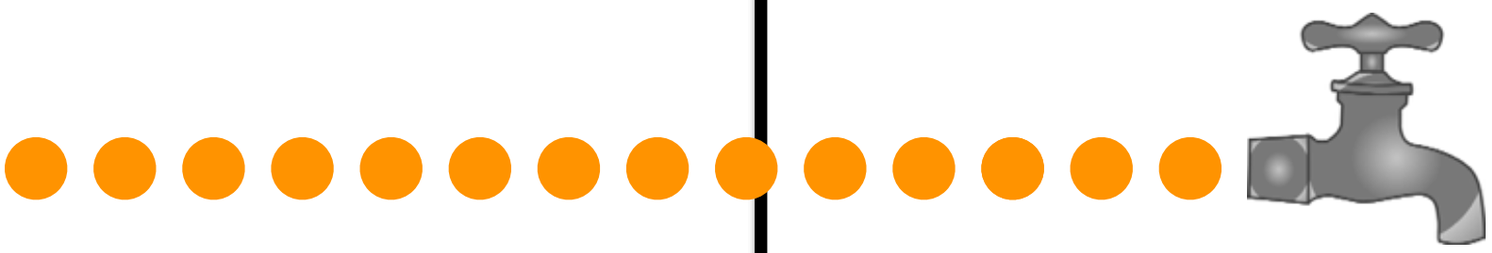


short-circuiting

intermediate operation

intermediate operation

terminal operation



stateful

sorted()

short-circuiting

intermediate operation

intermediate operation

terminal operation

see also:

Stream<T>

DoubleStream


LAZY
Stream
ONE TIME
LAZY
Guava
FluentIterable
MANY TIMES

Guava

Optional

~~Optional.map()~~

~~Optional.flatMap()~~

Stream		Guava FluentIterable
	anyMatch	
	limit	
count		size
findFirst		first
flatMap		transformAndConcat
distinct		toSet
sorted		toSortedList
Collectors.groupingBy		index

Guava ListenableFuture

map

flatMap

wrapping constructor

map

flatMap

Stream.of(value)

Optional.of(value)

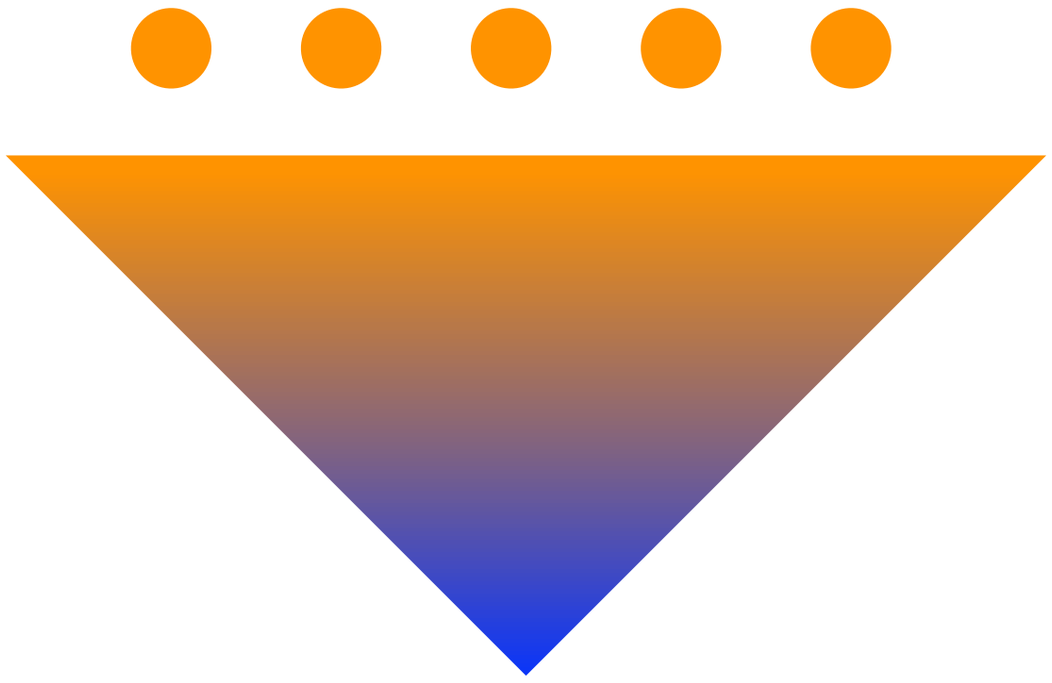
map

flatMap

wrapping

constructor

collect



also known as fold

Functions as Values

STORE functions in variables

PASS functions in parameters

RETURN functions from other functions

Remove duplication

PASS functions in parameters

Stream processing

Contexts