Using the Stream API to Map, Filter, and Reduce Data

José Paumard
PHD, JAVA CHAMPION, JAVA ROCK STAR

@JosePaumard https://github.com/JosePaumard

Agenda

Let us see the Stream API is action!

Let us create streams

And map / filter them

And also flat map them

Processing Data with Java Streams

```
List<Person> people = ...;

long count =
    people.stream()
          .map(person -> person.getAge()) // Function
          .filter(age -> age > 20)        // Predicate
          .count();
```

Calling stream() opens a stream on a collection

The map() method takes a Function

The filter() method takes a Predicate

count() is a terminal method on the Stream interface

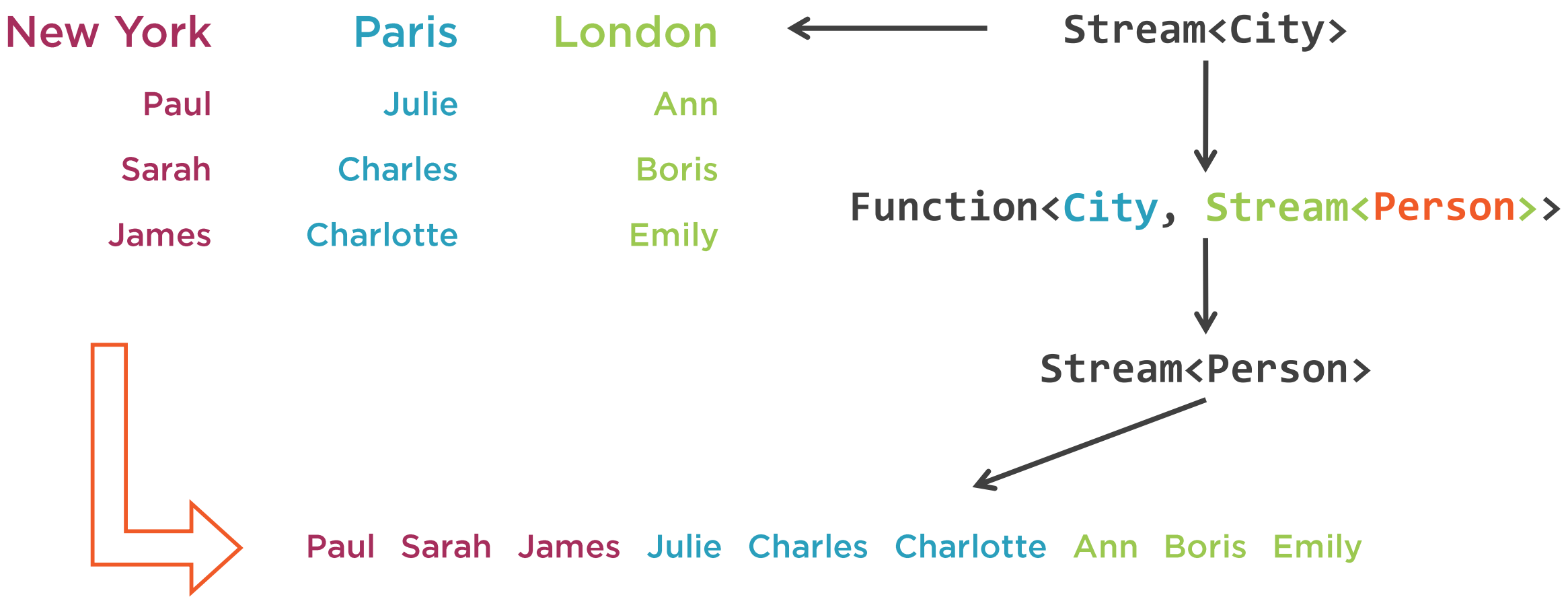
Demo

Let us write some code!

Run some map / filter / reduce

Flat Mapping Data

Flat Mapping a 1:p Relation



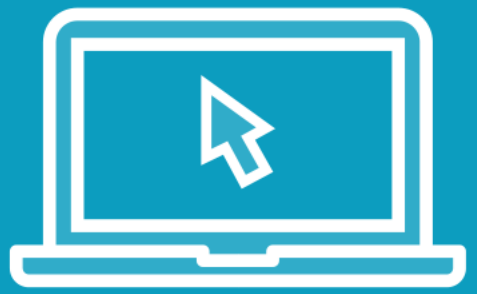
```
List<City> cities = ...;

Function<City, Stream<Person>> flatMapper =
    city -> city.getPeople().stream();

long count =
    cities.stream()                 // Stream<City>
          .flatMap(flatMapper) // Stream<Person>
          .count();
```

A flat map streams the 1:p relation of a stream of objects

It uses a flat mapper = a function that returns a stream

The flat map operation
is defined by a function
that takes an object
and returns a Stream of other objects

Demo



Let us write some code!

And see this flatMap() in action

Module Wrap Up

What did you learn?

First Streams in action!

Basic map / filter / reduce operations

Terminal operations: count and forEach

Flat map operator for 1:p relations