STREAMS EM JAVA

DANIEL ABELLA
www.daniel-abella.com

```java
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

public class StreamExample {

    public static void main(String[] args) {

        List<String> nomes = new ArrayList<>();
        nomes.add("Daniel");
        nomes.add("Nathaly");
        nomes.add("Arthur");
        nomes.add("Lucas");

        // Filtrando elementos com base em um predicado
        List<String> nomesFiltrados = nomes.stream()
                .filter(nome -> nome.startsWith("D"))
                .collect(Collectors.toList());

        System.out.println(nomesFiltrados);  // Saída: [Daniel]

        // Mapeando elementos para outro tipo
        //Convertendo para uma List de Int com o Tamanho de Cada String
        List<Integer> tamanhosNomes = nomes.stream()
                .map(String::length)
                .collect(Collectors.toList());

        System.out.println(tamanhosNomes);  // Saída: [6, 7, 6, 5]

        // Agrupando todas Strings em uma só
        String concatenacaoNomes = nomes.stream()
                .reduce("", (nome1, nome2) -> nome1 + " " + nome2);

        System.out.println(concatenacaoNomes);  // Saída: Daniel Nathaly Arthur Lucas
    }
}
```

1. `nomes.stream()` converte o ArrayList nomes para um stream.
`.filter(nome -> nome.startsWith("D"))` usamos o filter para filtrar apenas os elementos da stream que atendem a uma condição (neste caso, nome comece com a letra "D").
`.collect(Collectors.toList())` O método collect, como diz o método, coleta os elementos filtrados em uma nova lista. Este método é usado em conjunto com Collectors.toList() para agrupar os elementos filtrados em uma nova lista (neste caso, nomesFiltrados).

2. `nomes.stream()` converte o ArrayList nomes para um stream.
`.map(String::length)` O método map aplica uma transformação em cada elemento da stream. Neste exemplo, aplicamos a transformação String::length, que retorna o tamanho (número de caracteres) de cada nome da lista.
`.collect(Collectors.toList())` O método collect, como diz o método, coleta os elementos filtrados em uma nova lista. Este método é usado em conjunto com Collectors.toList() para agrupar os tamanhos dos nomes em uma lista (neste caso, tamanhoNomes).

3. `nomes.stream()` converte o ArrayList nomes para um stream.
`.reduce("", (nome1, nome2) -> nome1 + " " + nome2)` O método combina todos os elementos da Stream em um único resultado. O primeiro argumento `""` é o valor inicial do resultado, e o segundo argumento `(nome1, nome2) -> nome1 + " " + nome2`, é uma função Lambda que define a operação de redução. Neste caso, a função lambda concatena os nomes nome1 e nome2 com um espaço entre eles.