

## Pro Java №15

Streams (потоки) в Java предоставляют мощный и удобный набор классов и методов для работы с коллекциями данных.

Одной из основных идей функционального программирования является использование чистых функций, которые не изменяют состояния и не сохраняют результаты вычислений в переменных.

Метод `stream()` позволяет преобразовать коллекцию, такую как список, в последовательный поток данных, который можно легко обрабатывать с помощью различных операций потоков.

Чтобы поток (Stream) в Java выполнил свои операции, ему необходима терминальная операция. Промежуточные операции, такие как `filter`, `map` и `sorted`, возвращают новый поток и не выполняются до тех пор, пока не будет вызвана терминальная операция. Терминальная операция завершает работу потока и запускает выполнение всех промежуточных операций.

### Промежуточные операции (Intermediate Operations):

1. `filter`: Фильтрует элементы на основе заданного условия.
2. `map`: Преобразует каждый элемент потока с помощью заданной функции.
3. `flatMap`: Преобразует каждый элемент в поток и объединяет все потоки в один.
4. `sorted`: Сортирует элементы потока.
5. `distinct`: Удаляет дубликаты элементов из потока.
6. `limit`: Ограничивает количество элементов в потоке.
7. `skip`: Пропускает первые `n` элементов потока.
8. `peek`: Выполняет действие для каждого элемента потока без изменения потока.

### ### Конечные операции (Terminal Operations):

1. `**`forEach`**`: Выполняет действие для каждого элемента потока.
2. `**`collect`**`: Сбор элементов потока в коллекцию или другую конечную структуру данных.
3. `**`reduce`**`: Свертка элементов потока в одно значение, используя бинарную функцию.
4. `**`count`**`: Возвращает количество элементов в потоке.
5. `**`anyMatch`**`: Возвращает ``true``, если хотя бы один элемент соответствует условию.
6. `**`allMatch`**`: Возвращает ``true``, если все элементы соответствуют условию.
7. `**`noneMatch`**`: Возвращает ``true``, если ни один элемент не соответствует условию.
8. `**`findFirst`**`: Возвращает первый элемент из потока, если он есть.

Одним из ключевых аспектов работы с потоками (`Stream`) в Java является их неизменяемость. Это означает, что исходная коллекция, из которой был создан поток, остается неизменной.