# Top 10 Java Stream Methods with Examples

## 1. filter()

Purpose: Keep only the elements that match a condition.

List<String> names = Arrays.asList("Dixith", "Ravi", "Sri", "Dixith");  
  
List<String> filtered = names.stream()  
 .filter(name -> name.startsWith("D"))  
 .collect(Collectors.toList());  
  
System.out.println(filtered); // [Dixith, Dixith]

## 2. distinct()

Purpose: Remove duplicates.

List<String> names = Arrays.asList("Dixith", "Ravi", "Dixith");  
  
List<String> uniqueNames = names.stream()  
 .distinct()  
 .collect(Collectors.toList());  
  
System.out.println(uniqueNames); // [Dixith, Ravi]

## 3. map()

Purpose: Transform each element (change format, case, value, etc.)

List<String> names = Arrays.asList("dixith", "ravi");  
  
List<String> upperNames = names.stream()  
 .map(String::toUpperCase)  
 .collect(Collectors.toList());  
  
System.out.println(upperNames); // [DIXITH, RAVI]

## 4. sorted()

Purpose: Sort the elements (ascending by default)

List<Integer> numbers = Arrays.asList(3, 1, 4, 2);  
  
List<Integer> sorted = numbers.stream()  
 .sorted()  
 .collect(Collectors.toList());  
  
System.out.println(sorted); // [1, 2, 3, 4]

## 5. limit(n)

Purpose: Take only the first n elements.

List<String> names = Arrays.asList("A", "B", "C", "D");  
  
List<String> firstTwo = names.stream()  
 .limit(2)  
 .collect(Collectors.toList());  
  
System.out.println(firstTwo); // [A, B]

## 6. skip(n)

Purpose: Skip the first n elements.

List<String> names = Arrays.asList("A", "B", "C", "D");  
  
List<String> afterSkip = names.stream()  
 .skip(2)  
 .collect(Collectors.toList());  
  
System.out.println(afterSkip); // [C, D]

## 7. collect()

Purpose: Collect the stream result into a list, set, map, etc.

List<String> names = Arrays.asList("QA", "Dev", "Tester");  
  
List<String> list = names.stream()  
 .collect(Collectors.toList()); // collect into List  
  
Set<String> set = names.stream()  
 .collect(Collectors.toSet()); // collect into Set

## 8. forEach()

Purpose: Perform an action (like print, log) on each element.

List<String> roles = Arrays.asList("QA", "Dev", "Admin");  
  
roles.stream().forEach(role -> System.out.println("Role: " + role));

## 9. anyMatch(), allMatch(), noneMatch()

Purpose: Check conditions.

List<Integer> numbers = Arrays.asList(2, 4, 6);  
  
// Check if any number is odd  
boolean anyOdd = numbers.stream().anyMatch(n -> n % 2 != 0);  
System.out.println(anyOdd); // false  
  
// Check if all numbers are even  
boolean allEven = numbers.stream().allMatch(n -> n % 2 == 0);  
System.out.println(allEven); // true

## 10. count()

Purpose: Count elements after filtering or transforming.

List<String> names = Arrays.asList("Dixith", "Dev", "QA", "Dev");  
  
long count = names.stream()  
 .filter(name -> name.equals("Dev"))  
 .count();  
  
System.out.println(count); // 2