Python Lists and Filter() vs Java Equivalents

1■■ What is a List in Python?

A list in Python is an ordered, mutable collection of elements — similar to an array in JavaScript or Java. It can hold different data types and allows dynamic resizing.

Example:

```
arr = [1, 2, 3, 4, 5]
print(arr[0]) # 1
```

2■■ The filter() Function in Python

The filter() function filters elements from an iterable based on a condition (function returning True/False). It returns an iterator, usually converted to a list.

Syntax: filter(function, iterable)

Example:

```
arr = [1, 2, 3, 4, 5, 6]
even = list(filter(lambda x: x % 2 == 0, arr))
print(even) # [2, 4, 6]
```

3■■ List Comprehension Alternative

A Pythonic way to achieve the same filtering is using list comprehension.

Example:

```
even = [x for x in arr if x % 2 == 0]
```

4■■ Java Equivalent (Streams API)

In Java 8+, the Stream API provides similar functionality using filter() and lambdas.

Example:

```
List arr = Arrays.asList(1, 2, 3, 4, 5, 6);
List even = arr.stream()
.filter(x -> x % 2 == 0)
.collect(Collectors.toList());
System.out.println(even); // [2, 4, 6]
```

5■■ Removing Falsy Values

Python:

```
arr = [0, False, None, '', 5, True]
filtered = list(filter(bool, arr))
print(filtered) # [5, True]
```

Java:

```
List arr = Arrays.asList(0, false, null, '', 5, true);
List filtered = arr.stream()
```

```
.filter(x -> x != null && !x.equals(false) && !x.equals('') && !x.equals(0))
.collect(Collectors.toList());
System.out.println(filtered); // [5, true]
```

6■■ Summary Table

Python vs Java Comparison:
• Python list → Java List<T>
• filter(fn, list) → stream().filter(fn)
• lambda x: cond → x -> cond
• list(filter(bool, arr)) → stream().filter(non-null)
• [x for x in arr if cond] → stream().filter(cond).collect()