

# Interview Questions

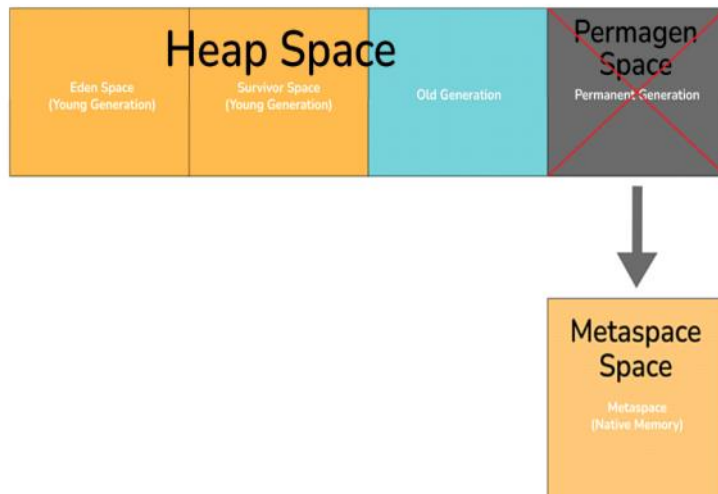
31 May 2022 20:32

## Java 8:

- What are the changes done to Java 8 interface ?
  - ◆ Introduction of Static and default methods
- What is the need or what is the advantage of having the default methods in interface in Java 8
  - ◆ Backward compatibility.
- Can we override default method of interface ?
- What is the advantage of static methods in Java 8 interface ?
  - ◆ Utility method or null check.
- How to call default method present in interface ?
  - ◆ `InterfaceName.super.methodName()`
- How to call static method present in interface ?
  - ◆ `InterfaceName.methodName()`
- What is functional interface ?
  - ◆ It's an interface which has 1 abstract method and it can have any number of static and default methods also it should be annotated with `@functional` Interface
- What is lamda and what are the advantages of Lamda ?
  - ◆ Lamda enables functional programming
  - ◆ Anonymous inner class can be replaced by lamda
  - ◆ Reduced code
- What is functional programming ?
  - ◆ Passing the functionality as method argument
- What happens when we write more than 1 abstract in interface for which already lamda expression is present (If the interface is annotated with `@functional` interface)?
  - ◆ Compilation error during writing the second abstract method
  - ◆ If there is no `@functional` interface annotation then it allows to write more than 1 abstract method but compilation at the place lamda expression
- What is the difference between interface and functional interface ?
  - ◆ Difference is only in number of abstract methods. Functional interface will have only 1 abstract method, where as interface can have any number of abstract methods.
  - ◆ Both can have any number of default and static methods
- What are the different functionals interfaces introduced in Java 8.
  - ◆ Predicate, Supplier, Consumer, Function
- What is Stream ?
  - ◆ `Java.util.Stream` -> it's an interface.

- What is the advantage of Stream ?
  - ◆ It can be used to perform operation on the collection.
- How to create the stream ?
  - ◆ `Stream.of(array)`
  - ◆ `CollectionObject.stream()`
  - ◆ `Stream.of(directly providing the array values)`
- What are the changes introduced in collection in Java 8 ?
  - ◆ `forEach`, `removeIf`
- Write a stream code to delete the duplicates from array List
  - ◆ Here List object is h-> `h.stream().distinct().collect(Collectors.toList())`
- Write a stream code to delete the duplicates from array list which has Department object
  - ◆ Here List object is h-> `h.stream().distinct().collect(Collectors.toList())` and override hash code and equals method
- Write a program to sort the Array List which has user defined data (Department Object)?
  - `h.stream.sorted((d,d1)->d.getDepId()  
<d1.getDepId()).collect(Collectors.toList())`
- Write a Stream program to convert the array into List
  - ◆ Array is `int[] x={10,11,2,4,5,6,7};`
  - ◆ `Stream.of(x).collect(Collectors.toList());`
- Write a Stream program to convert the Employee Array into Map (Map should have key as Emp id and value as Emp Object)
  - `Employee[] e={new Employee("Karthik",10),new  
Employee("Ashok",13)}`
  - `Stream.of(e).collect(Collectors.toMap(p->p.getId(),p->p))`
- What is the difference between map and flat Map ?
  - ◆ Map provides one to one mapping
  - ◆ `flatMap` provides one to many mapping
  - ◆ If you want to convert `List<List<Employee>>` to `List<employee>` - then u go for flat map
  - ◆ `List<Employee>` -> `EmpId` as key and value as emp Object then go for Map
- Write a stream program to get minimum, maximum, average salary of an employee (given employee array)
  - ◆ `l.stream().min(comparator).get()` -> Min salary
  - ◆ `l.stream().max(comparator).get()` -> Max salary Or
  - ◆ `l.stream().collect(Collectors.summarizingInt())`
  - ◆ `DoubleSummaryStatistics d  
=empList.stream().collect(Collectors.summarizingDouble((e)->e.getEmpSalary()));`
- What is difference between intermediate operation and terminal operation
  - ◆ Intermediate operation doesn't provide output until terminal operation is called
- What is short circuit operation ?

- ◆ It comes under terminal operation and based on the value it stops the execution immediately
- What is the difference forEach and forEachOrdered ?
  - ◆ forEach is not ordered in case of parallel stream
  - ◆ forEachOrdered is ordered in case of parallel Stream
  - ◆ For the foreachOrdered method if the input is ordered object output is ordered else it can follow any order
- What are the examples of intermediate operation ?
  - ◆ Filter, map, sorted, distinct
- What are the examples of terminal operation ?
  - ◆ forEach, collect, min, count, max
- What is the difference between limit and skip methods of Stream ?
  - ◆ Limit will limit the elements
  - ◆ Skip will skip elements and do the operation afterwards
- What is optional ?
  - ◆ Optional is a class in Java 8 which is introduced to overcome null pointer exception
- How to create the optional object ?
  - ◆ Optional.ofNullable(null or not null value):
  - ◆ Optional.ofNullable(null)
  - ◆ Optional.ofNullable(10)
  - ◆ Optional.ofNullable(new Employee("karthgik"))
- What is Local Date Time or Local Date or Local Date ?
  - ◆ New Date and time objects in java 8
- How to get current date and time using the LocalDateTime class ?
  - ◆ LocalDateTime.now()
- Array List consists of 5 employees and Write a lambda to sort the employee by last name.
  - ◆ Collections.sort(l,(e1,e2)->e1.getLastName().compareTo(e2.getLastName()));
- Print all the employees whose name starts with c ?
  - ◆ l.stream().filter(p->p.startsWith("c")).forEach(p->sysout(p))
- Print all the employees using the enhanced collections in Java 8
  - ◆ forEach
- What is MetaSpace? How does it differ from PermGen?



**PremGen:** MetaData information of classes was stored in PremGen (Permanent-Generation) memory type before Java 8. PremGen is fixed in size and cannot be dynamically resized. It was a contiguous Java Heap Memory.

**MetaSpace:** Java 8 stores the MetaData of classes in native memory called 'MetaSpace'. It is not a contiguous Heap Memory and hence can be grown dynamically which helps to overcome the size constraints. This improves the garbage collection, auto-tuning, and de-allocation of metadata

➤ What are functional or SAM interfaces?

- ◆ Functional Interfaces are an interface with only one abstract method. Due to which it is also known as the Single Abstract Method (SAM) interface. It is known as a functional interface

➤ Can a functional interface extend/inherit another interface?

- ◆ A functional interface cannot extend another interface with abstract methods as it will violate the rule of one abstract method
- ◆ It can extend only marker interface (or) if it extends any other functional interface which has 1 abstract method then in child we should not write any abstract method
- ◆ Out of n interfaces if we are performing inheritance then only 1 abstract method should be present

➤ It can extend other interfaces which do not have any abstract method and only have the default, static, another class is overridden

➤ Given a list of integers, find out all the even numbers exist in the list using Stream functions?

- ◆ `List<Integer>myList  
=Arrays.asList(10,15,8,49,25,98,32);myList.stream().filter(n ->n%2==  
0).forEach(System.out::println);`

- **Given a list of integers, find out all the numbers starting with 1 using Stream functions?**

- ◆ `List<Integer>myList  
=Arrays.asList(10,15,8,49,25,98,32);myList.stream().map(s ->s + "")//  
Convert integer to String.filter(s ->  
s.startsWith("1")).forEach(System.out::println);`

- **Given a list of integers, find the total number of elements present in the list using Stream functions?**

- ◆ `List<Integer>myList  
=Arrays.asList(10,15,8,49,25,98,98,32,15);longcount  
=myList.stream().count();System.out.println(count);`

- **Given a list of integers, find the maximum value element present in it using Stream functions?**

- ◆ `List<Integer>myList =Arrays.asList(10,15,8,49,25,98,98,32,15);intmax  
=myList.stream().max(Integer::compare).get();System.out.println(max);`

- **Write a Lambda to Sort the given numbers ?**

- ◆ `List<Integer> list = Arrays.asList(57, 38, 37, 54, 2);  
list.stream()  
.sorted()  
.forEach(System.out::println);`

- **Write a Program to sort the elements in descending order ?**

- ◆ `List<String> strings =  
Arrays.asList("Stream","Operations","on","Collections");  
strings.stream()  
.sorted( (s1, s2) -> s2.length() - s1.length() )  
.forEach(System.out::println);  
◆ Strings.stream().sorted(collections.reverseOrder()).forEach()`

- **How to write the method reference for a Static method ?**

`ClassName::MethodName`

- **How to write method reference for instance method ?**

- ◆ `ObjectName::MethodName`

- **How to write method reference for constructor ?**

- ◆ `ClassName::New`

- **How to create local date with values ?**

- ◆ `Of()` method of `LocalDate`

- Is the Local date class immutable ?
  - ◆ yes
- What is the advantage of `orElse()` or `OrElseGet()` method of Optional class ?
  - ◆ If we want to replace the Null with a specific String value then we go for `OrElse()`
  - ◆ You can get the default value using `OrElseGet(supplier)`
- What is the difference between `findAny()` and `findFirst()` ?
  - ◆ If it is ordered then first element will be returned by `findFirst` or `findAny`
  - ◆ If it is not ordered output can be anything.
- Write a Program to sort hashmap by keys using Stream ?
  - ◆ Refer to the code mentioned in telegram group
- Write a program to Sort hashmap by values using Stream ?
  - ◆ Refer to the code mentioned in telegram group

## Java 9

- What is Jshell ?
  - ◆ Jshell is Repl tool
- Why Java 9 doesn't have `rt.jar` ?
  - ◆ Because from Java 9 onwards java follows modular approach
- What is the difference between module and Jar ?
  - ◆ Module contains extra file called `module-info.java`
- What principle does Jshell follow ?
  - ◆ REPL: Read, Evaluate, Print, Loop
- What are the advantages of Jshell ?
  - ◆ Developer friendly
- What is snippet in Jshell ?
  - ◆ Any valid java statement, variable declaration or method, or interface or class
- What is the command to know all the snippets in Jshell
  - ◆ `/list -all`
- What is scratch variable ?
  - ◆ It is something which is create by Jshell
- What is the command to see all the methods in Jshell ?
  - ◆ `/methods`
- How to delete the method or variable in Jshell ?
  - ◆ `/drop snippetId`
- What is inactive snippet ?
  - ◆ `/list - Active snippets`
  - ◆ `/list -all - inactive`
- How to list all active and inactive snippets ?
- What is the command to see all the interfaces and classes present in Jshell ?

- ◆ /types
- How to edit the content in Jshell ?
  - ◆ /edit
- How Many modules present in Jdk 9 ?
  - ◆ 98
- What is the base module in Jdk 9 ?
  - ◆ Java.base
- What does module-info.java contains ?
  - ◆ Module description, exports, requires etc
- Explain the project architecture with multiple module ?
  - ◆ Look at the example code
- How to access one module code in another module ?
  - ◆ Exports packageName in 1 module and which ever module is using then it should requires PackageName
- What is the difference between requires and requires transitive ?
  - ◆ Requires transitive-> compile time
  - ◆ Requires both compile and run time
- What is the difference between Qualified export and exports ?
  - ◆ Export PackageName -> will be visible for all the modules
  - ◆ Export packageName to module1, module2 -> this package will be visible for only module 1 and module 2 packages
  - ◆ To reduce the visibility of a module we go for Qualified exports.
- What is try with resource ?
  - ◆ Generally we close the resource in finally block, but with the help of try with Resource JVM will close the resource automatically without mention finally block
- What is the change performed in try with resource in Java 1.9 ?
  - ◆ Local variable reference can be accepted in Java 1.9 try() block
  - ◆ Bufferread r1, bufferReader r2
  - ◆ Try(r1,r2) -> This is possible in java 1.9
- What is the change performed for diamond operator in java 1.9 ?
  - ◆ We can use diamond operator for Anonymous innerclass
- What is takeWhile() ?
  - ◆ Same as Limit method but the difference is limit will take number and takewhile will accept predicate -> if performs the operation till the condition is satisfied. Once the condition is failed it will stop
- What is dropWhile() ?
  - ◆ Same as Skip() but the difference is skip() will take number and dopwhile will take predicate -> it ignores the data till the predicate is satisfied once fails then does the operation
- Write a Stream program to display the employees whose salary is <10000 using takeWhile()
  - ◆ empList.Stream.takeWhile(p->p.getSalary>10000).forEach() ->
- What is the difference between takeWhile and filter?
  - ◆ Filter will execute through out the stream

- ◆ Takewhile will execute till the predicate is true. Once the predicate fails then it will stop the execution and doesn't apply that condition to the rest of the stream.
- Write a program to create the immutable collection object without Java 9?
  - ◆ `List<Integer> c= new ArrayList<Integer>();`
  - ◆ `c.add(10);c.add(20);`
  - ◆ I want make C as immutable object ->`Collections.unmodifiableList();`
- Write a program to create the immutable collection object with Java 9?
  - ◆ `List.of(arguments);`
  - ◆ `Set.of(arguments)`
- What happens if you perform add or remove on immutable collection ?
  - ◆ Runtime Exception-> UnsupportedOperationException
- Is it possible to declare same package in different modules ?
  - ◆ No.
- What is the advantage of modularity or module-info.java in Java 9 ?
  - ◆ Security,
  - ◆ Jar size will be reduced.
  - ◆ JVM will check for dependencies before start of the program so there is no way we get no class def error from Java 9
- Write a program to close the specific task ?
  - ◆ Process API->
  - ◆ `ProcessHandler pr=ProcessHandler.of(taskId)`
  - ◆ `Pr.destory();`
- Write a program to display all the running processes ?
  - ◆ `ProcessHandler pr=ProcessHandler.allProcesses()`
- Write a program to open eclipse using java program or start the process using java 9 Process API?
  - ◆ `ProcessBuilder("exepath").start()`
- What is the advantage of having the private methods in interface ?
  - ◆ Reusability of code present in interface
- Can you access interface private method outside of interface ?
  - ◆ No

## Java 11

- Can you execute a program without compilation ?
  - ◆ Yes from java 11, we can execute the program using the below command and it does the compilation internally
  - ◆ `Java className.java` (Example: `java Test.java`)
- What is the difference between `trim()` and `strip()` method in Java 11
  - ◆ `strip()` method uses `Character.isWhitespace()` method to check if the character is a whitespace. This method uses Unicode code points whereas `trim()` method identifies any character having codepoint value less than or equal to 'U+0020' as a whitespace character



- What are the changes to predicate in Java 11 ?
  - ◆ Not method
- What are the file changes
  - ◆ readString and Write String
- What are the String API changes
  - ◆ isBlank(), isEmpty(), Strip(), leadingString(), trailingStrip(), lines()
- What are the collection API changes ?
  - ◆ toArray()
- What is local variable var and how can we use it in lambda ?