**Java 8 Interview Questions**

**What are the java 8 new features?**  
  
This is the most asked interview question on Java 8. Many of the java developers do not work on java 8. Below are the java 8 features:  
  
**1. Functional Interface:** Each functional interface has a single abstract method, called the functional method, implementation can be provided using the lambda expressions.  
**2. Lambda Expressions:** It is a feature derived from functional programming. It is a function that does not belong to any class.  
**3. Optional:**Instead of using null values Optional class is used for representing Optional values.  
**4. Stream API**  
**5. Spliterator**  
**6. Method References**  
**7. New Date and Time API.**

**Differences between Collection API and Stream API are as follow :**  
  
1. Collection API was introduced in JDK 1.2 while Stream API is introduced in JDK 1.8  
2. Collection objects are created eagerly while Stream API objects are created lazily.

**What is Lambda Expression?**  
According to Oracle docs,  
  
Lambda expressions are the method without name i.e Anonymous method. In other words, Lambda expression is a function that can be passed around and referenced as an object.

A lambda expression consists of three parts :  
  
a. Parameter List  
b. Lambda symbol ->  
c. Expression

(Parameter List) ->{expression;}

**What is the difference between PermGenSpace and MetaSpace?**  
  
In JDK 8 onwards PermGenSpace is removed. Earlier PermGenSpace is used for storing the metadata. Metadata means storing information about classes like bytecodes, names, and JIT information.  
Java classes metadata now stored in a native heap and this space is called MetaSpace. Metaspace grows automatically by default and will be garbage collected.  
  
So the major difference between PermGenSpace and MetaSpace is that PermGenSpace was fixed in size and did not grow automatically, but MetaSpace does not have any size constraints.

**What is Functional Interface in Java 8?**  
  
In simple words, the Functional interface has exactly one abstract method. A compile-time error is thrown if an interface declaration is annotated with @FunctionalInterface  but is not.

A functional interface can contain default and static methods which do have an implementation.

**What do you understand by the term SAM interface?**  
  
Java 8 provided the feature of a functional interface. Since a functional interface can contain only one abstract method, hence, they are called SAM or "Single Abstract Method".

**What is a default method in Java 8? When to use it?**  
The default method is also known as defender methods or virtual extension methods. It is a non-abstract method i.e have a body, which can be declared inside the interface.  
The default method is introduced in Java 8 for backward compatibility. That is if you add a new abstract method to the interface, all the implementing classes shall break. Implementing classes need to implement the added abstract method. This problem is solved by the default method of java 8.

**What is the difference and similarities between Function and Predicate in java 8?**  
  
**Difference:**  
 **1. Return Type:**Function returns an Object and it is a single argument function.  
The predicate return type is boolean (i.e true or false) and it is also a single argument function.  
  
**Similarities:**  
  
**1.** Both are functional interfaces i.e both contain a single abstract method.

**Example of Lambda expression**

Runnable r1 = () -> {

for (int i = 0; i <= 5; i++) {

System.*out*.println("Thread1 " + i);

}

};

Runnable r2 = () -> {

for (int j = 0; j <= 5; j++) {

System.*out*.println("Thread2 " + j);

}

};

Thread t1 = new Thread(r1);

Thread t2 = new Thread(r2);

t2.start();

t1.start();

**Example of method Reference:**

interface interfaceDemo{

void add();

}

public class MethodRef {

public static void add1() {

System.*out*.println("Doing addition");

System.*out*.println("Doing addition again");

}

public void nonstaticMethd() {

System.*out*.println("Nonstatic method by method reference");

}

public static void main(String[] args) {

//static method ref

interfaceDemo ifDemo =MethodRef::*add1*;

ifDemo.add();

//lambda

interfaceDemo demo = ()-> System.*out*.println("lamda Expression");

demo.add();

//nonstatic method ref

MethodRef methodRef = new MethodRef();

interfaceDemo demo2 = methodRef::nonstaticMethd;

demo2.add();

}

}

**What is Stream API in Java 8? Why do we need it?**  
  
Stream API is the new feature of Java 8. It is used to process or compute the data.  
  
**Why do we need Stream API**  
  
a. Stream API supports aggregate operations that simplify the processing.  
b. It provides Functional-Style programming.