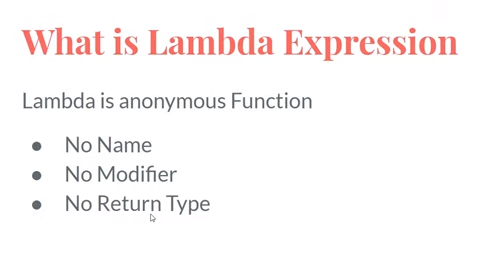
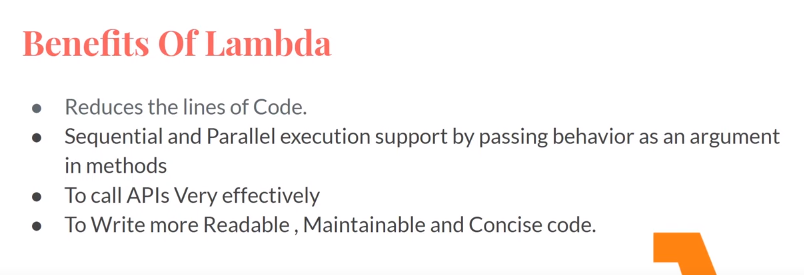
## What are Lambda Expressions?

Lambda expressions are another way of writing inline unnamed code block which performs a simple function. Not a direct analogy but if you recall anonymous classes in Java they are a way of writing inline classes. On similar lines Lambda expressions are a way to write method body inline.

They can be considered as anonymous methods, there is no need to formally define these methods but the code in the lambda expression is the body of the method.

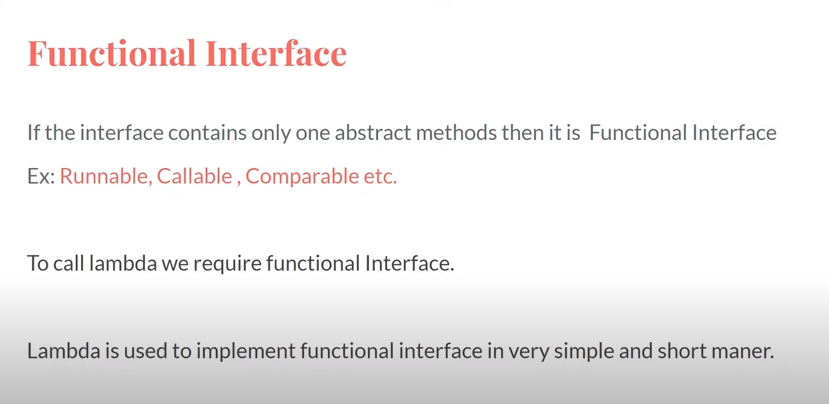


What is Functional Interface?

A functional interface is any interface which contains only one abstract method. A functional interface can have one or more default and static methods.

Also, there is an annotation in Java @FunctionalInterface which is used to annotate the functional interfaces. It is not a mandate for functional interfaces to carry this annotation.

There are many predefined functional interfaces in Java which belong to the package **java.util.function**



What is the significance of these interfaces?

These interfaces as we know declare a method which can be implemented and used for executing expressions as any other interface. But the power comes in conjunction to the [Lambda Expressions](http://techieme.in/introduction-to-lambda-expressions-java8/) we learnt in the previous post.

As the lambda expressions, are inline method definitions, it can serve as an implementation to functional interfaces.

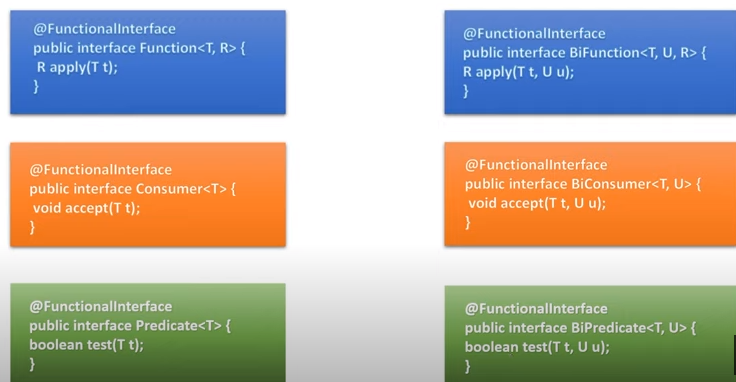
There are primarily four types of functional interfaces defined in the language:

* Function
  + It represents a relation which transforms an input T into a result R.
  + The functional method is **R** **apply(T t).**
* Consumer
  + It represents a relation which consumes an input T and returns no output.
  + The functional method is **void accept(T t).**
* Predicate
  + It represents a relation which evaluates an input T and produces a boolean result.
  + The functional method is **boolean test(T t).**
* Supplier
  + It represents a supplier of results
  + The functional method is **T get().**

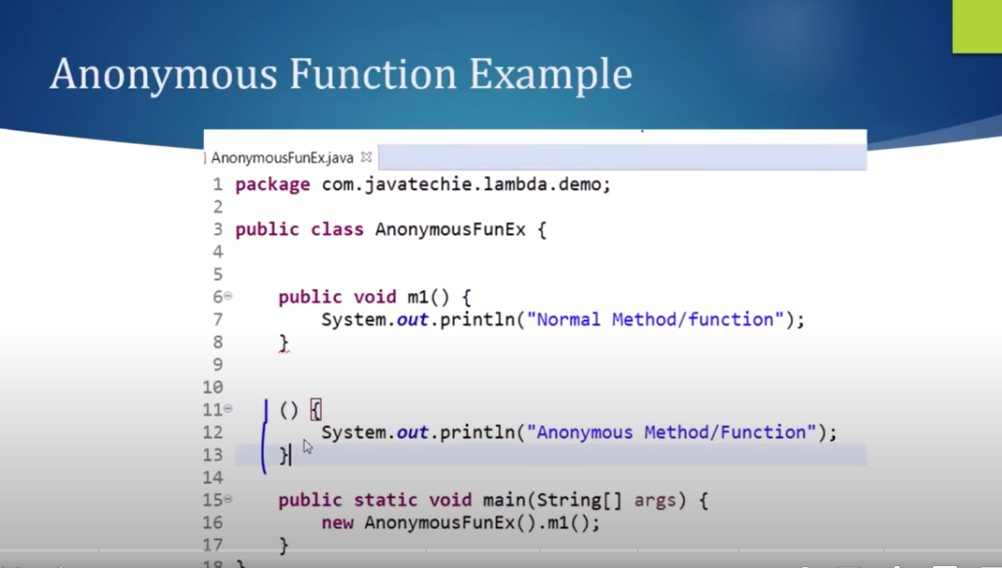
Apart from these methods there are certain default and static methods defined in these interfaces. Yes! you heard it right, you can define methods with bodies in Java 8.

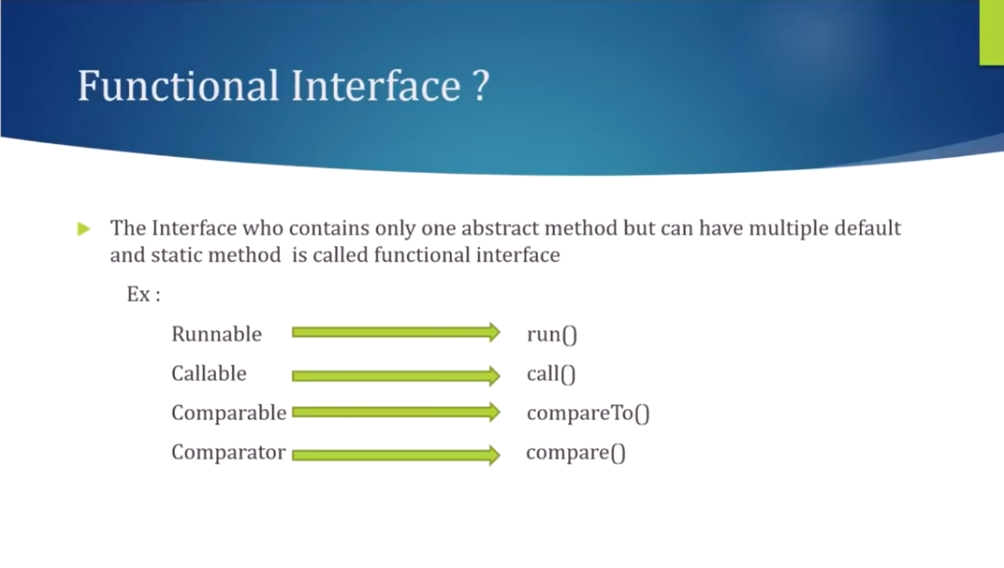
There are other categorization of the Functional Interfaces based on the number of arguments:

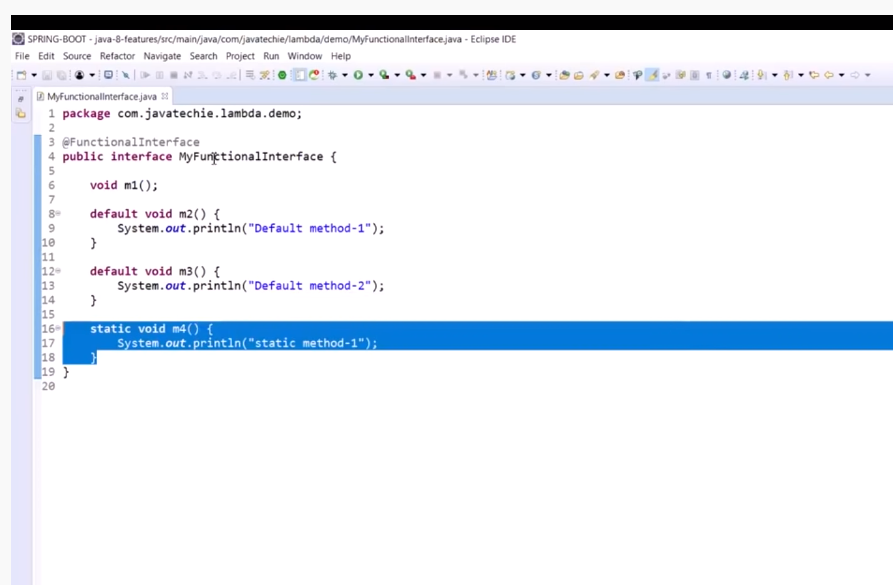
* BiPredicate
* BiConsumer
* BiFunction

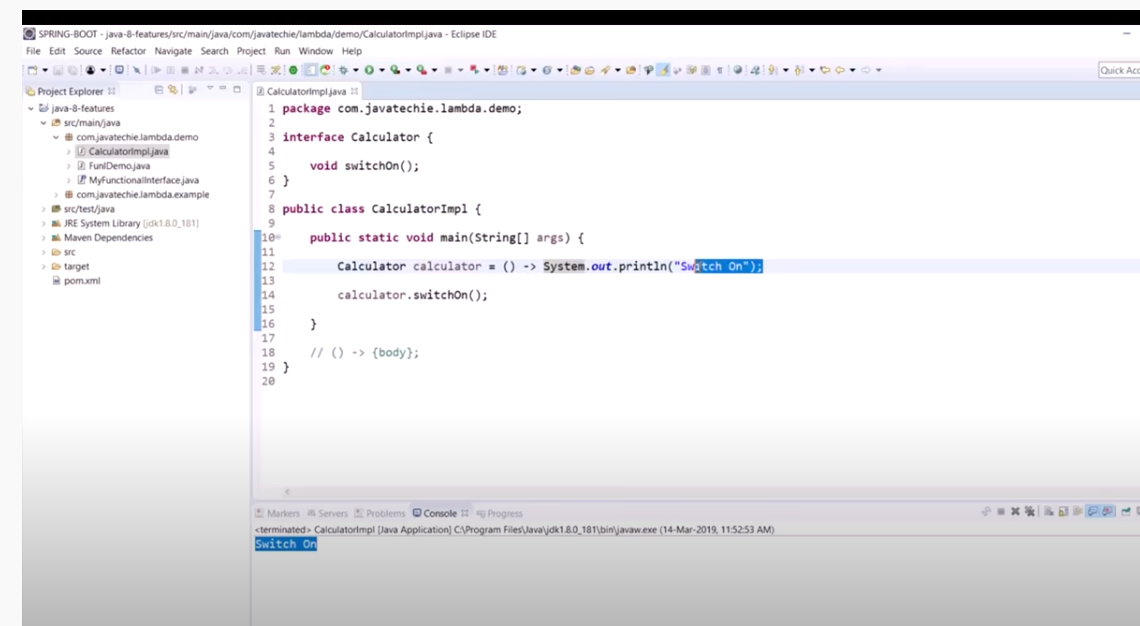


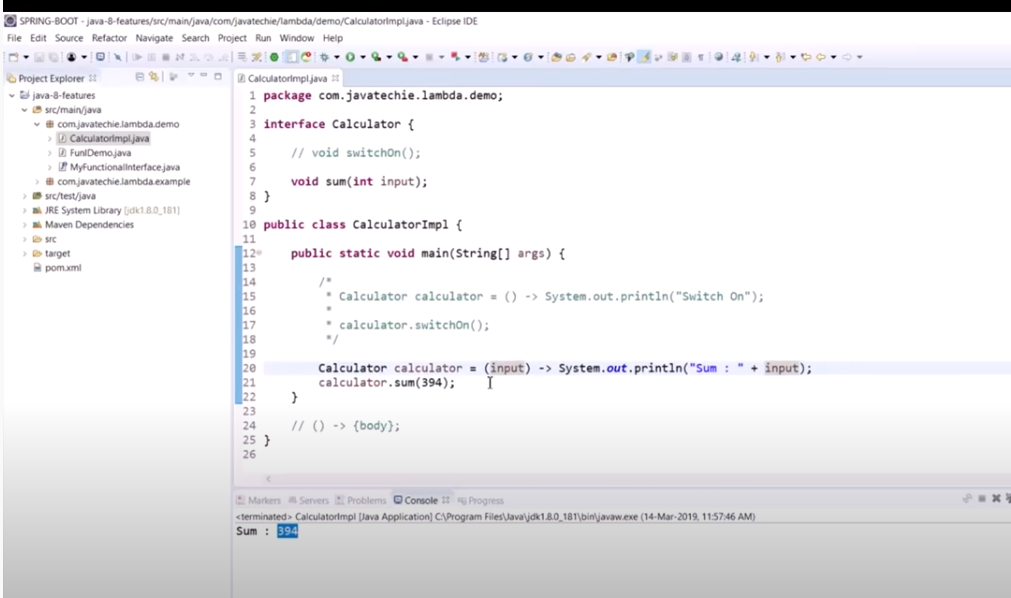
https://techieme.in/introduction-to-functions-java8/

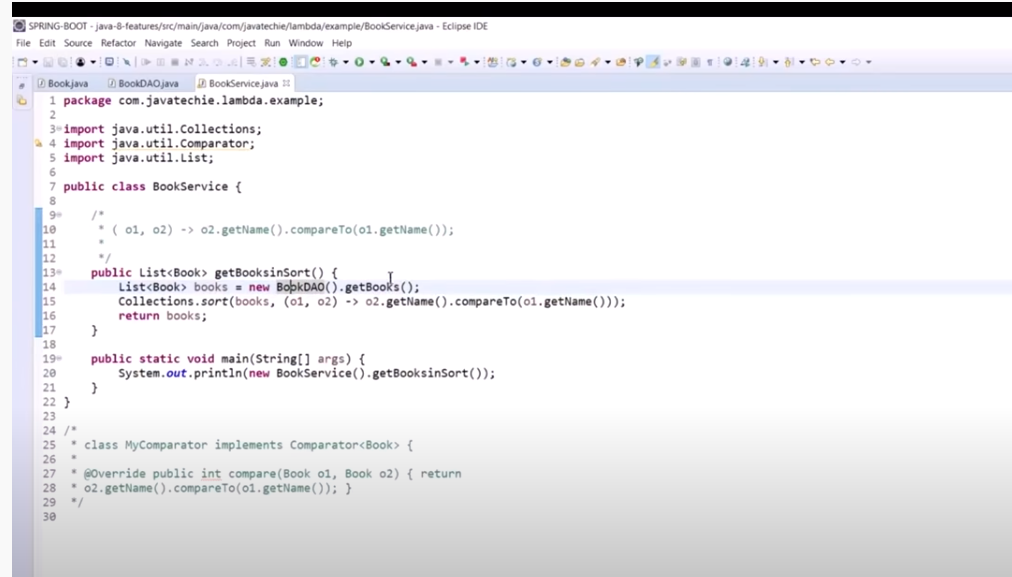




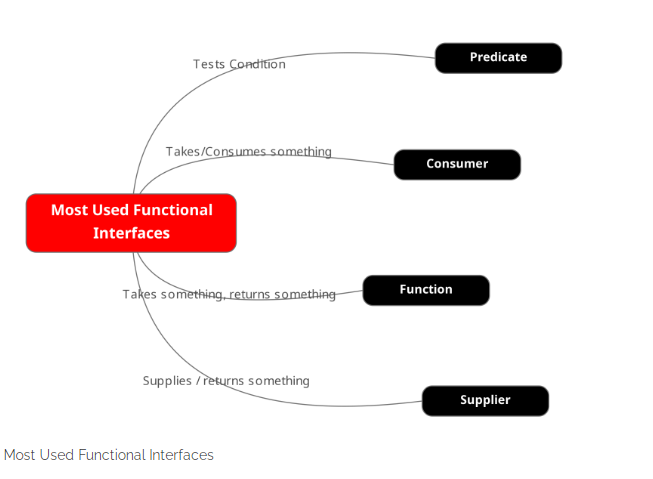
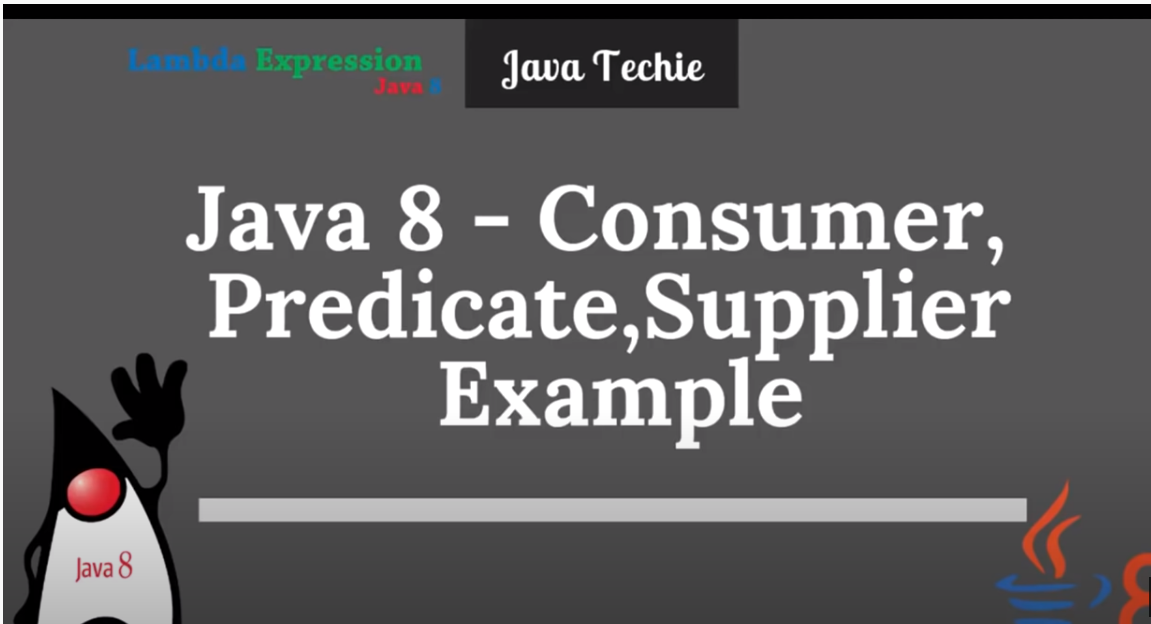








2nd Session



https://basicsstrong.com/understanding-predefined-functional-interfaces-in-java/

