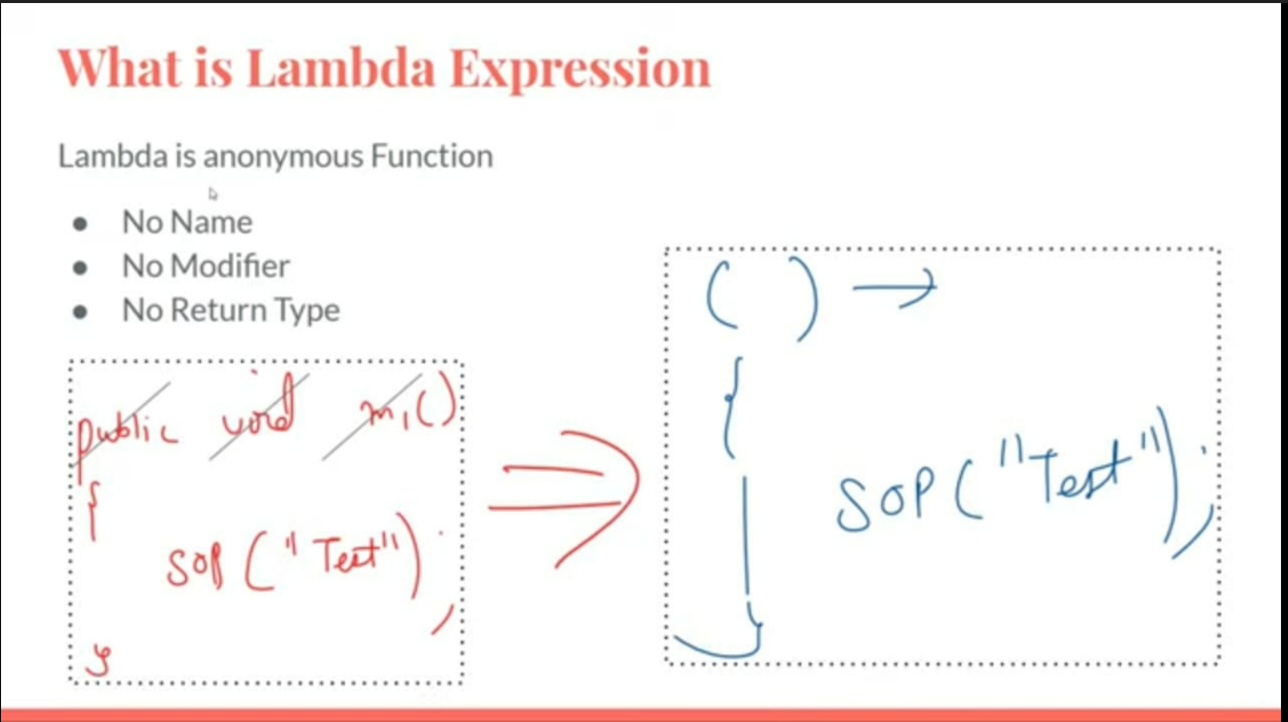
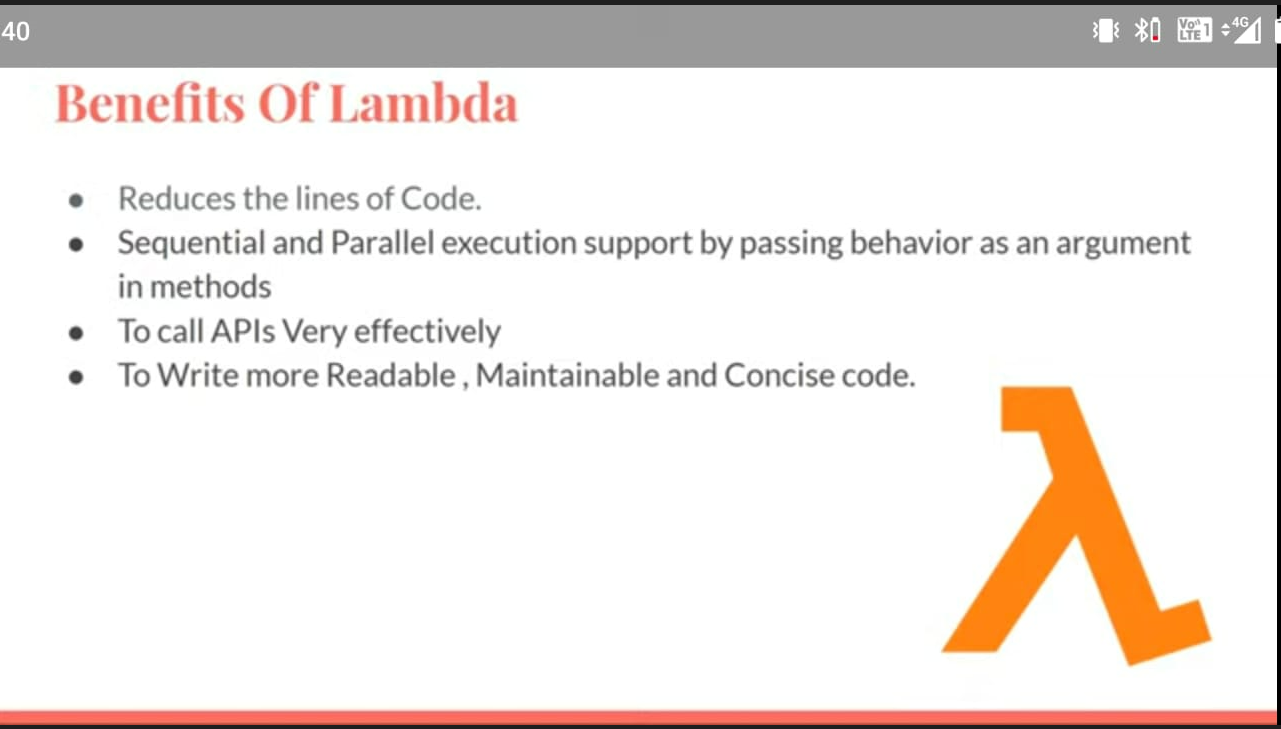
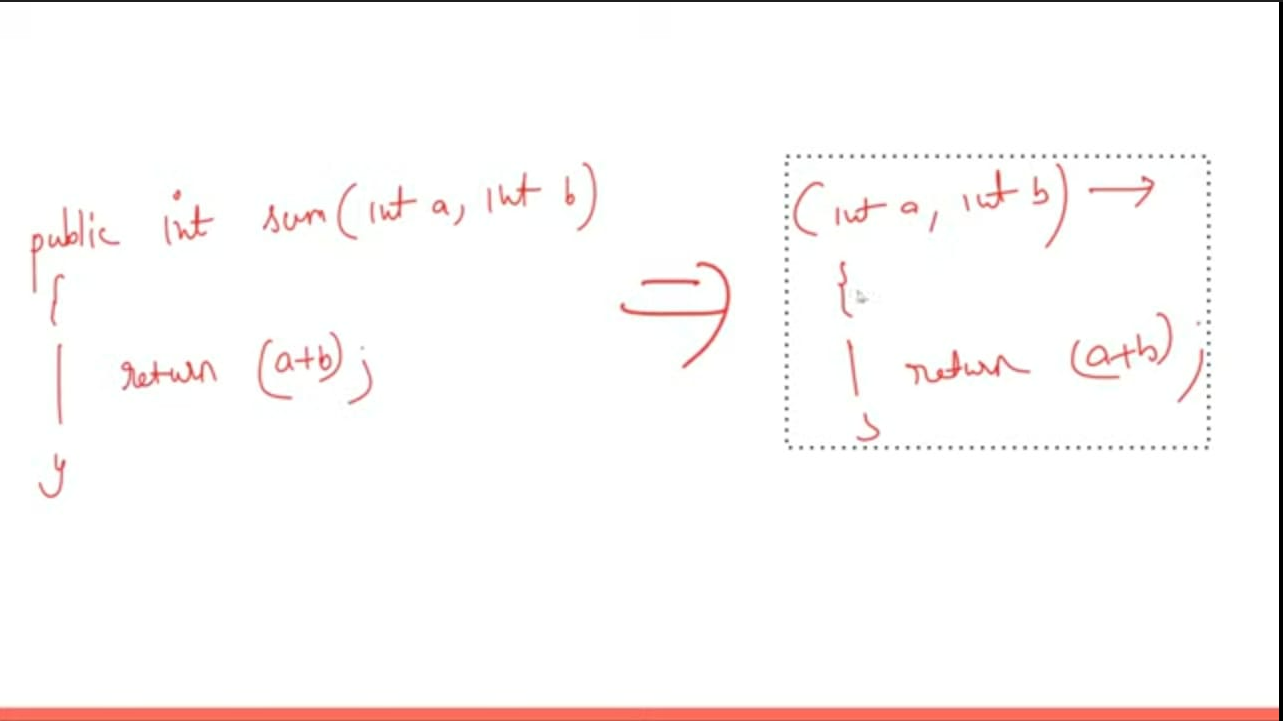
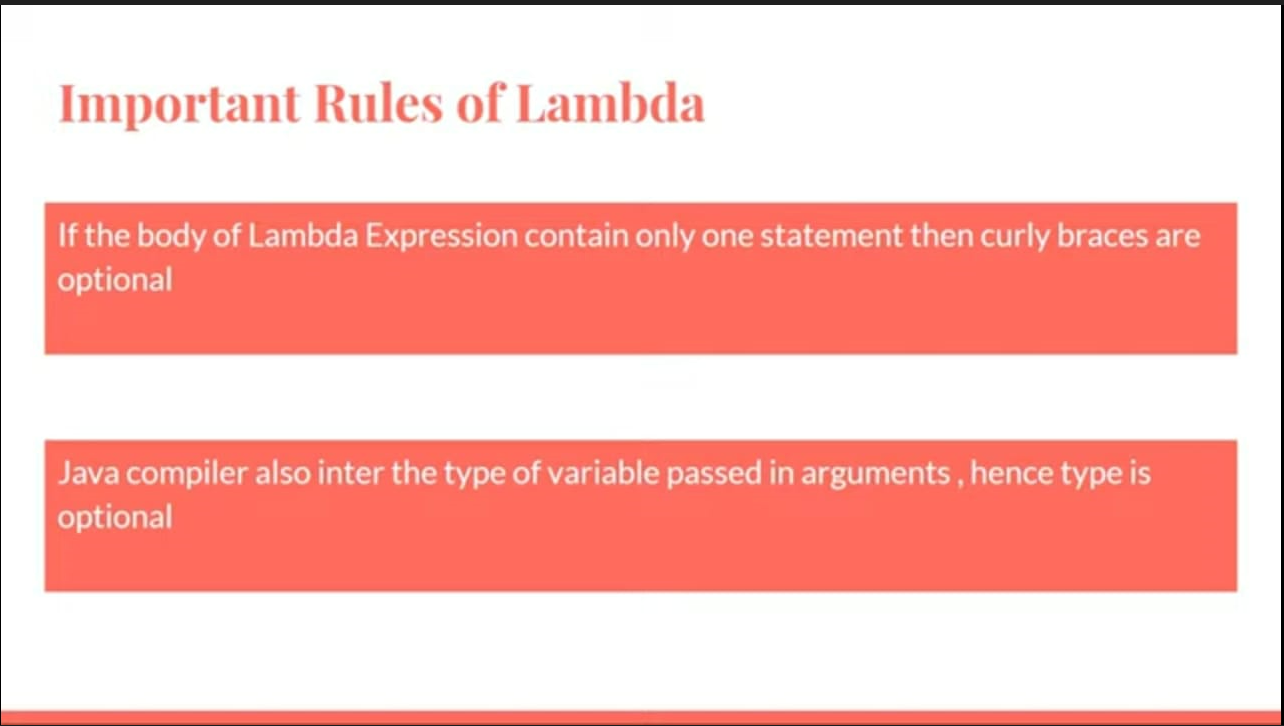
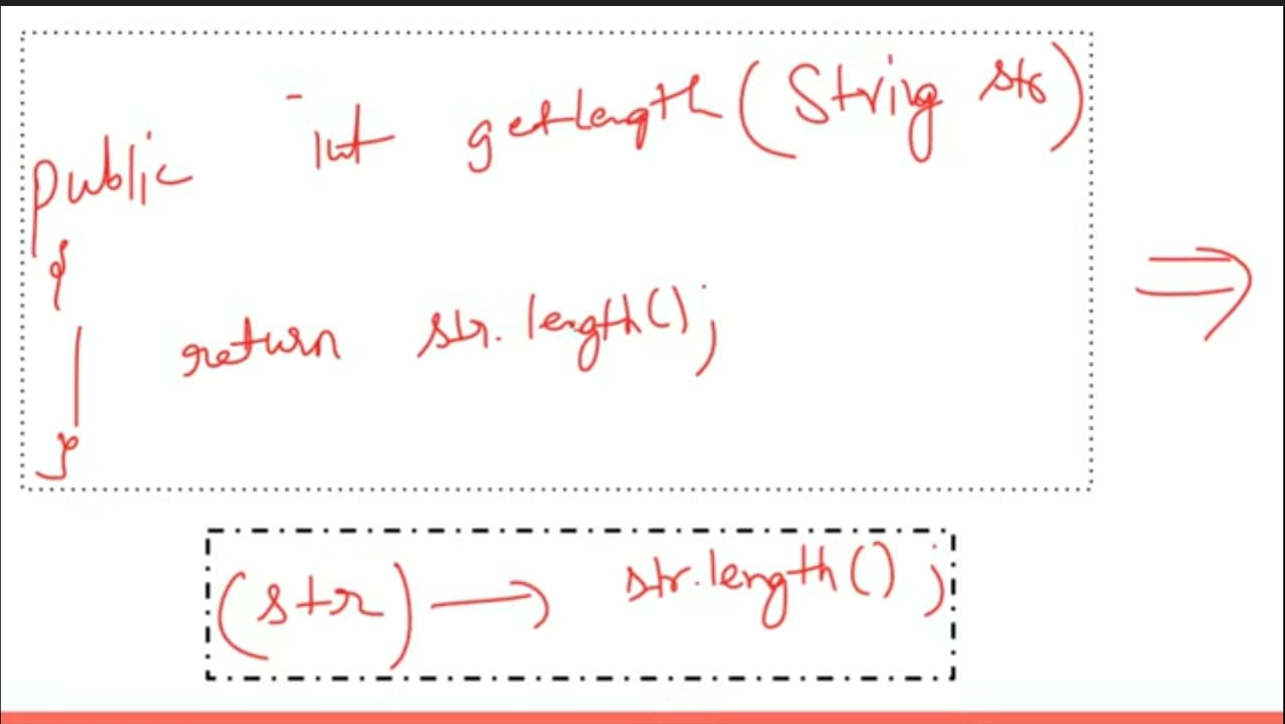
**Lambda**

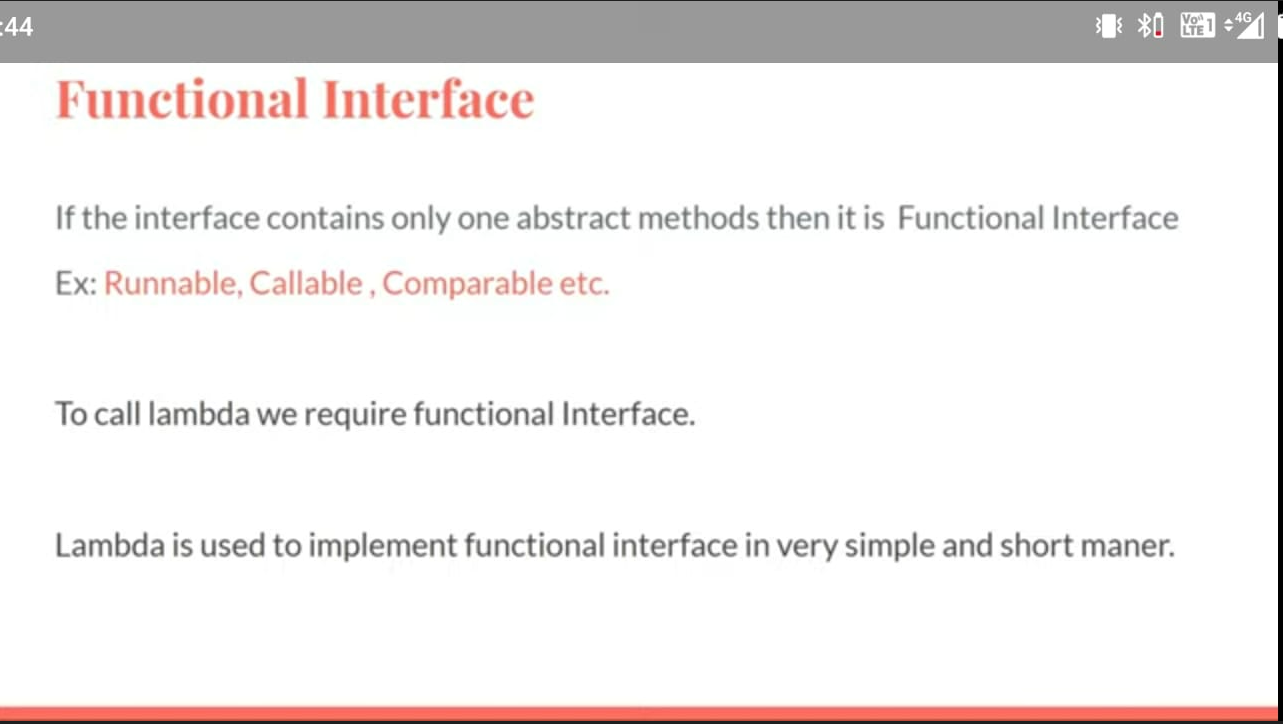




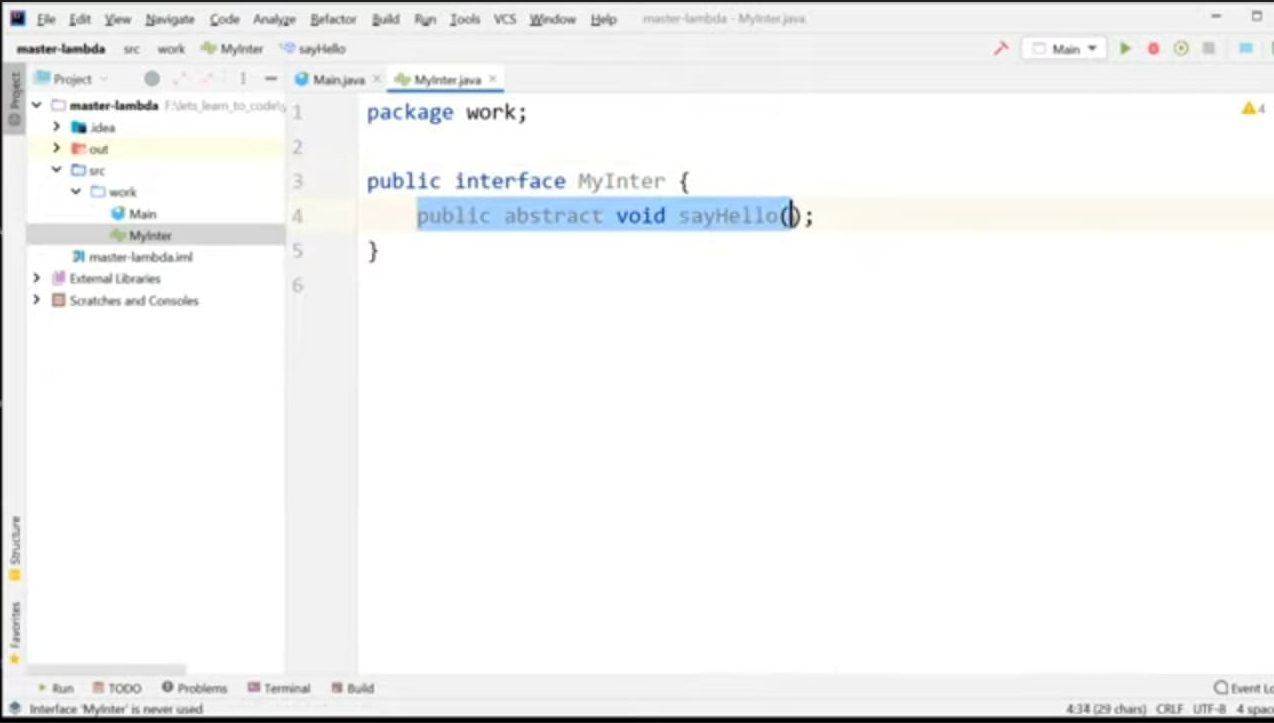


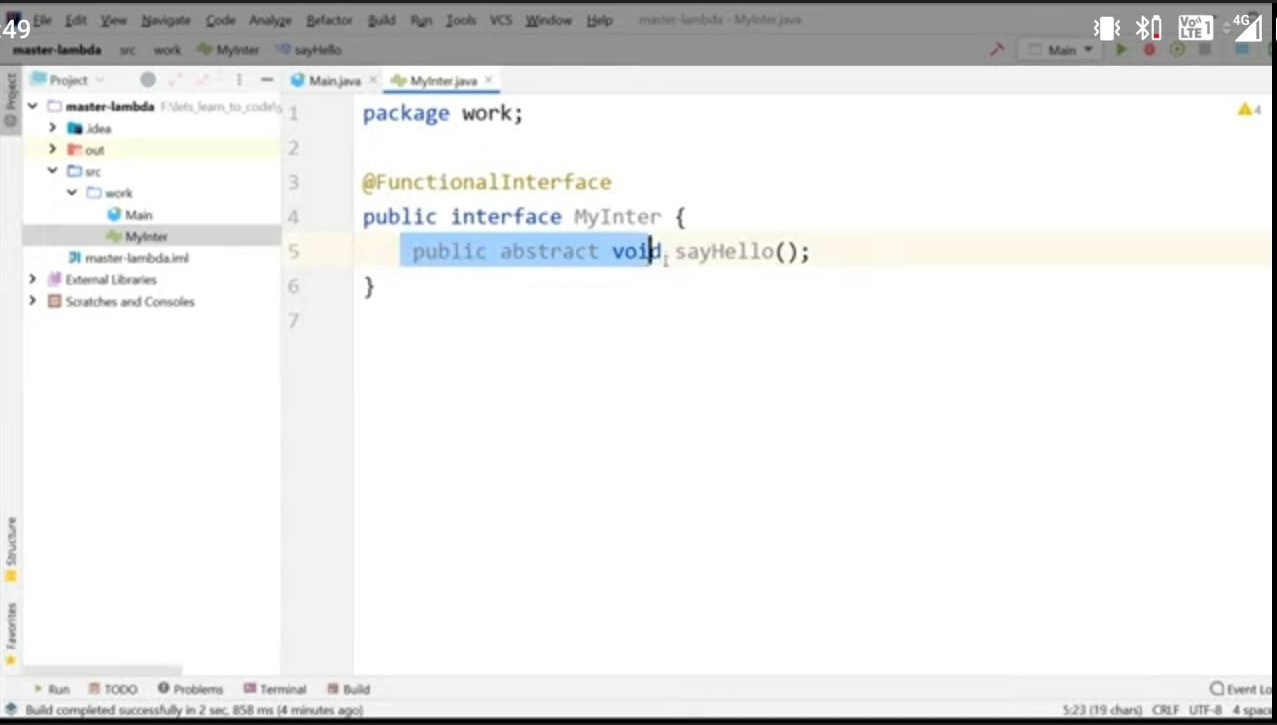




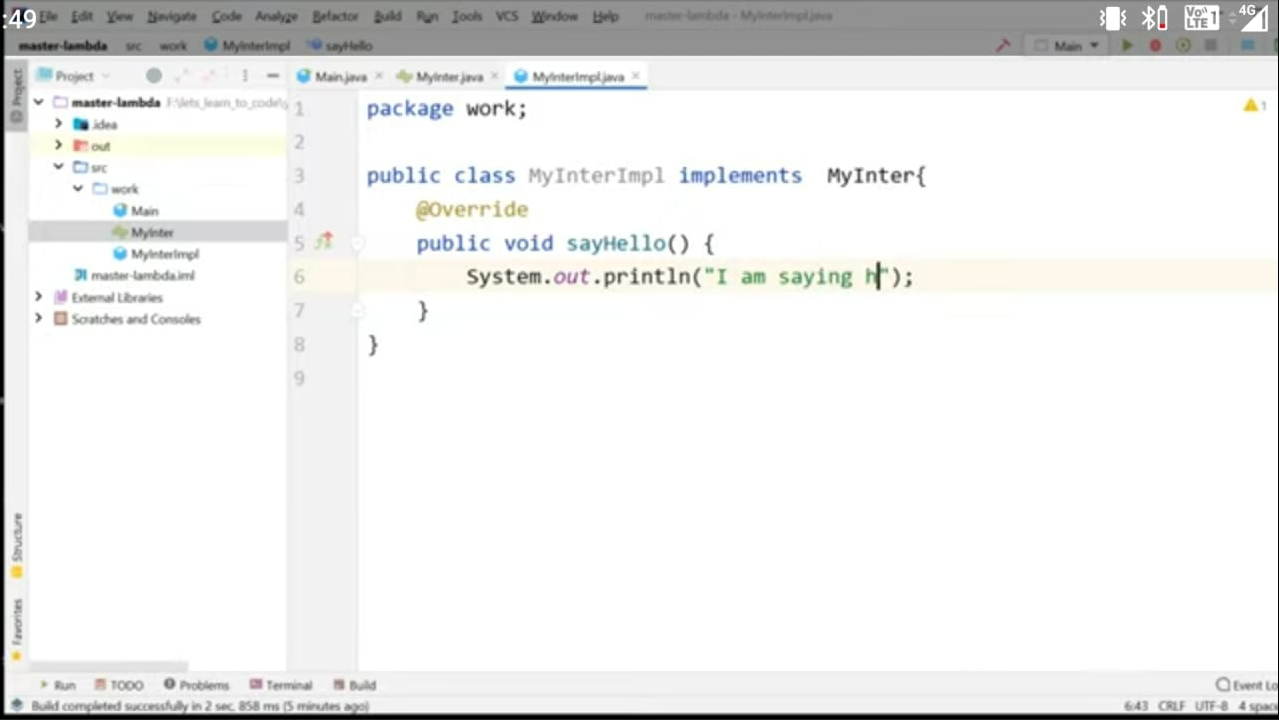




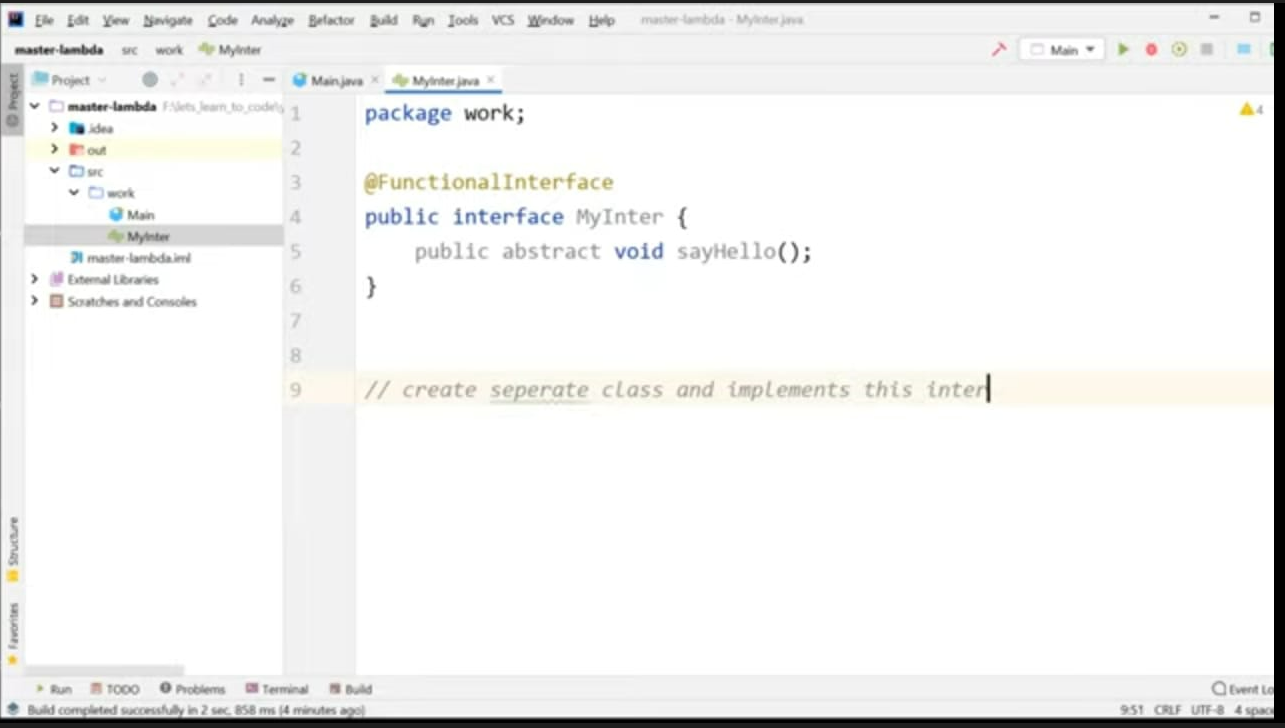


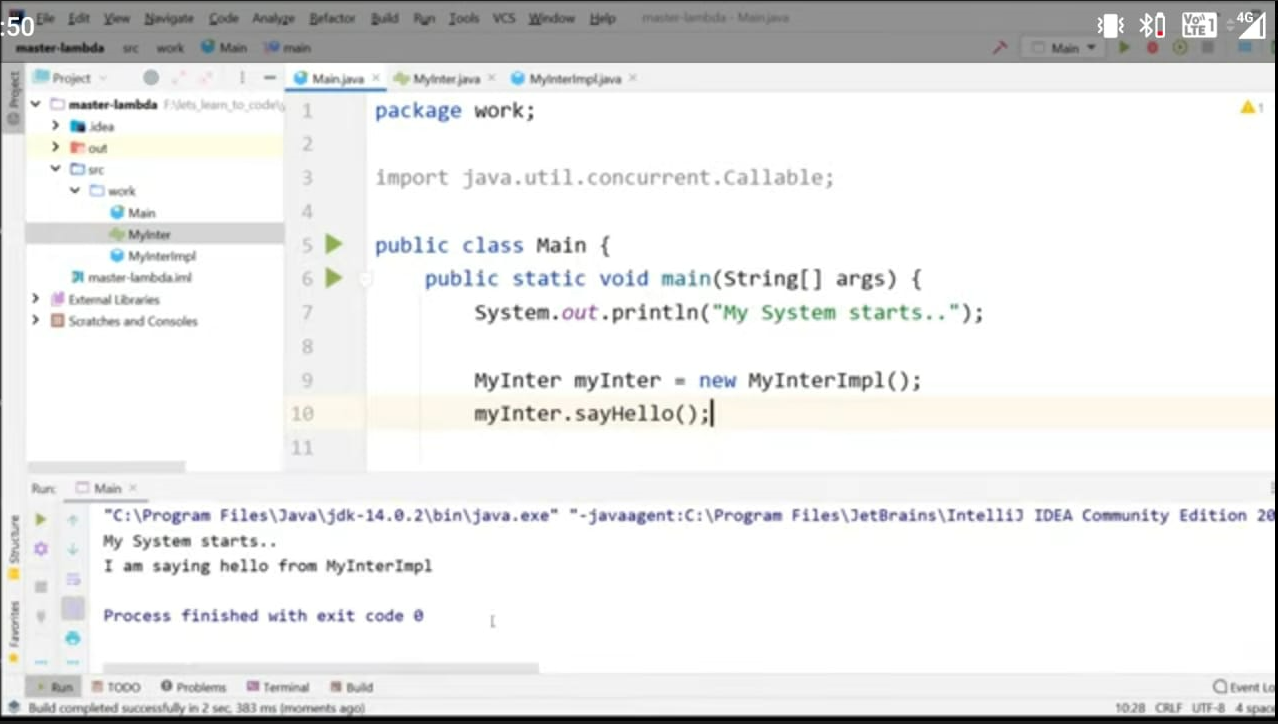


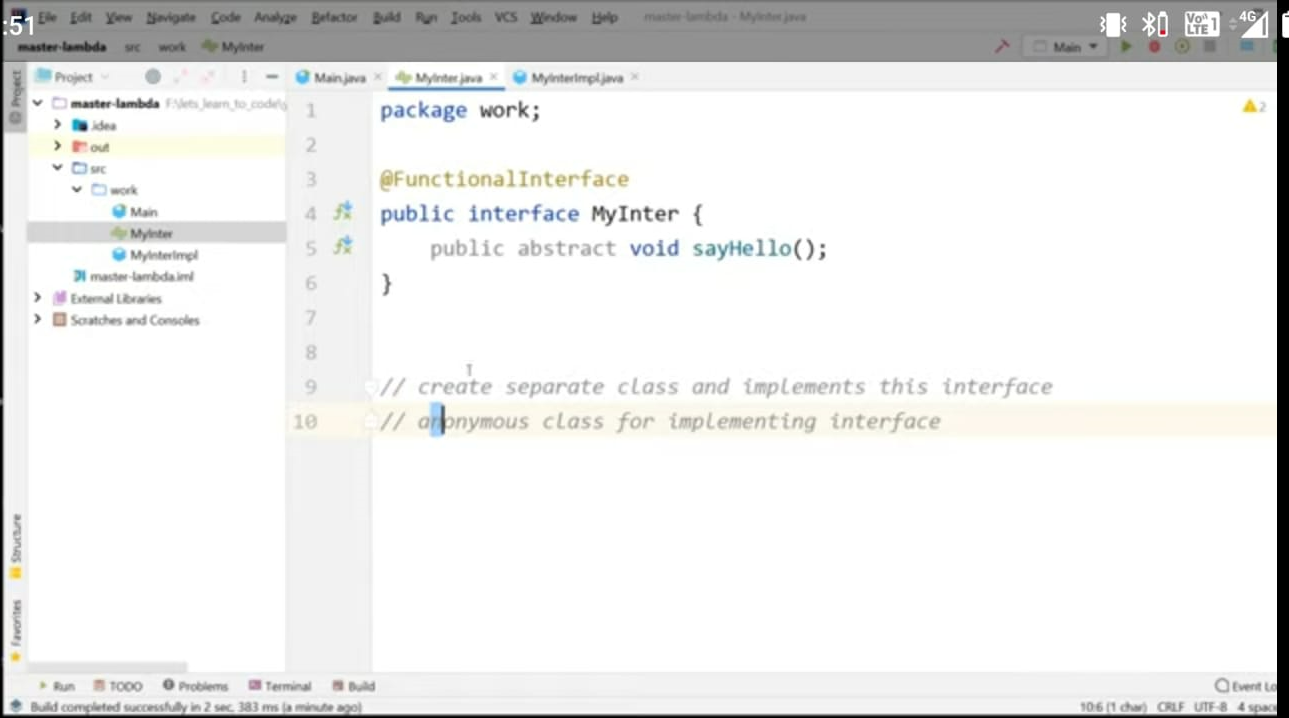
We can use annotation also to specify functional interface. If we don’t specify annotation and interface contains only one abstract method, then it is also treated as functional interface. Functional interface can have static and default method but should contain only one abstract method.



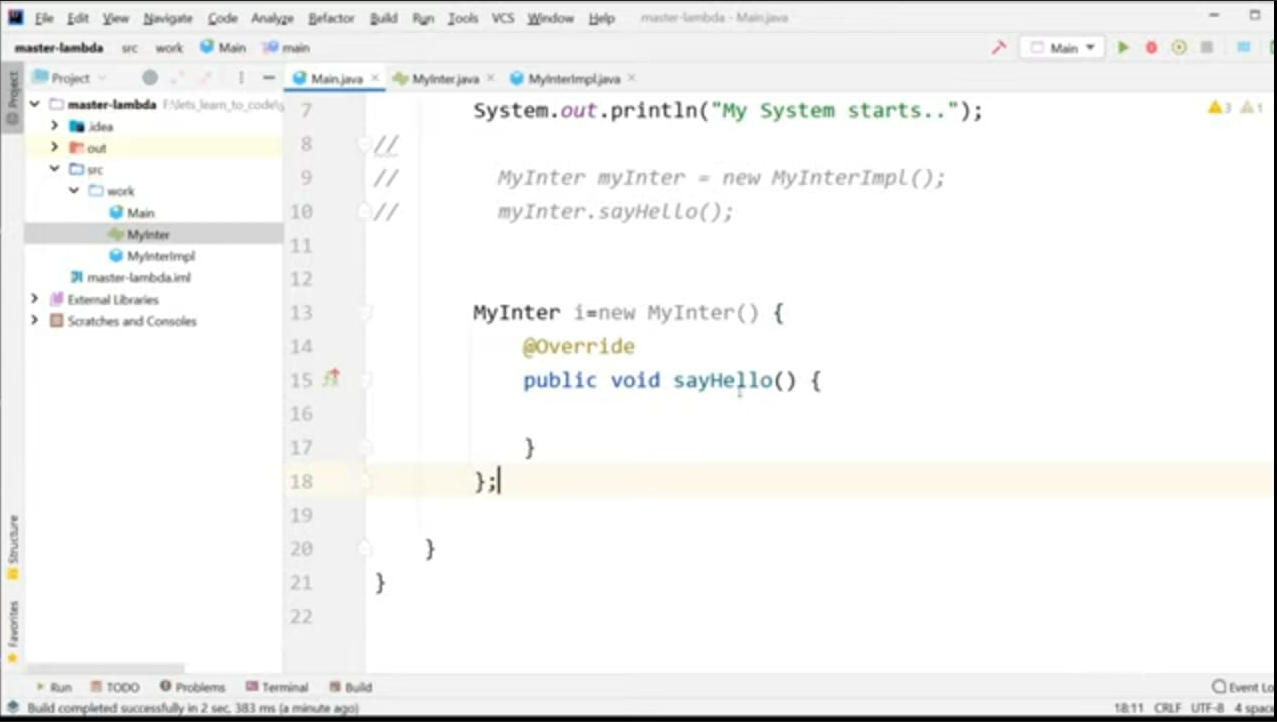
Traditional way to implement interface creating a class and implement interface.

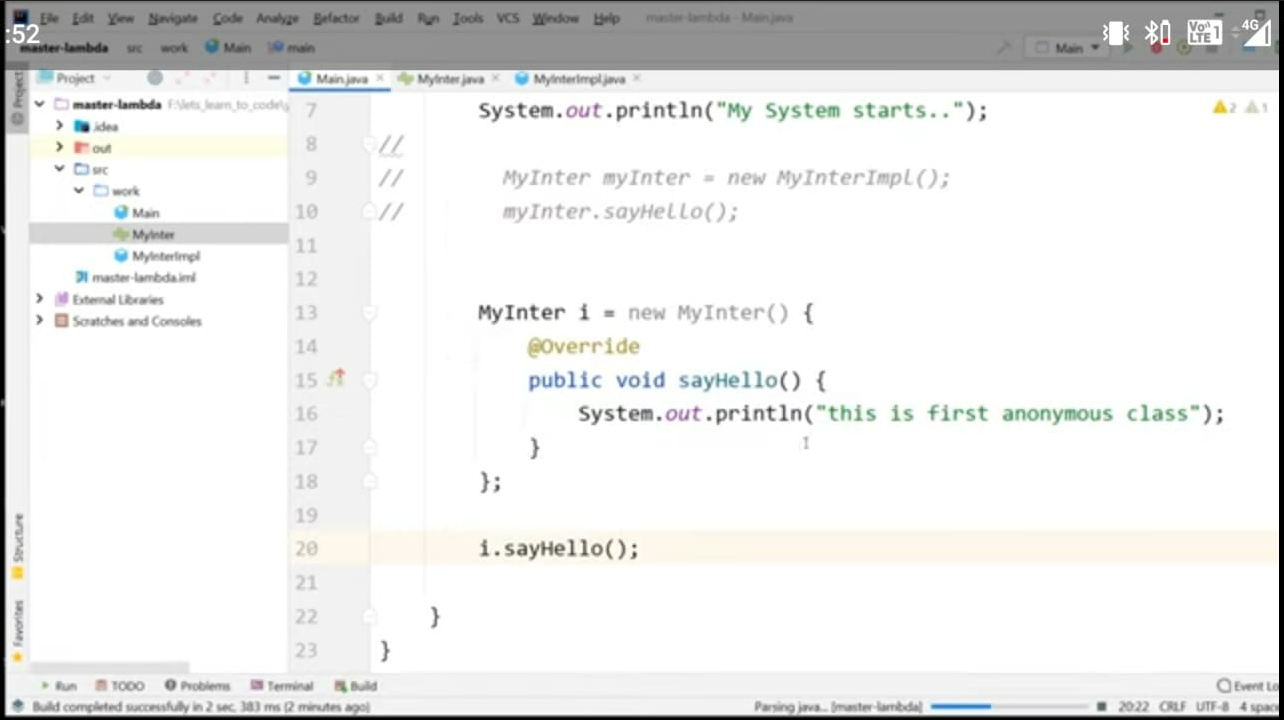


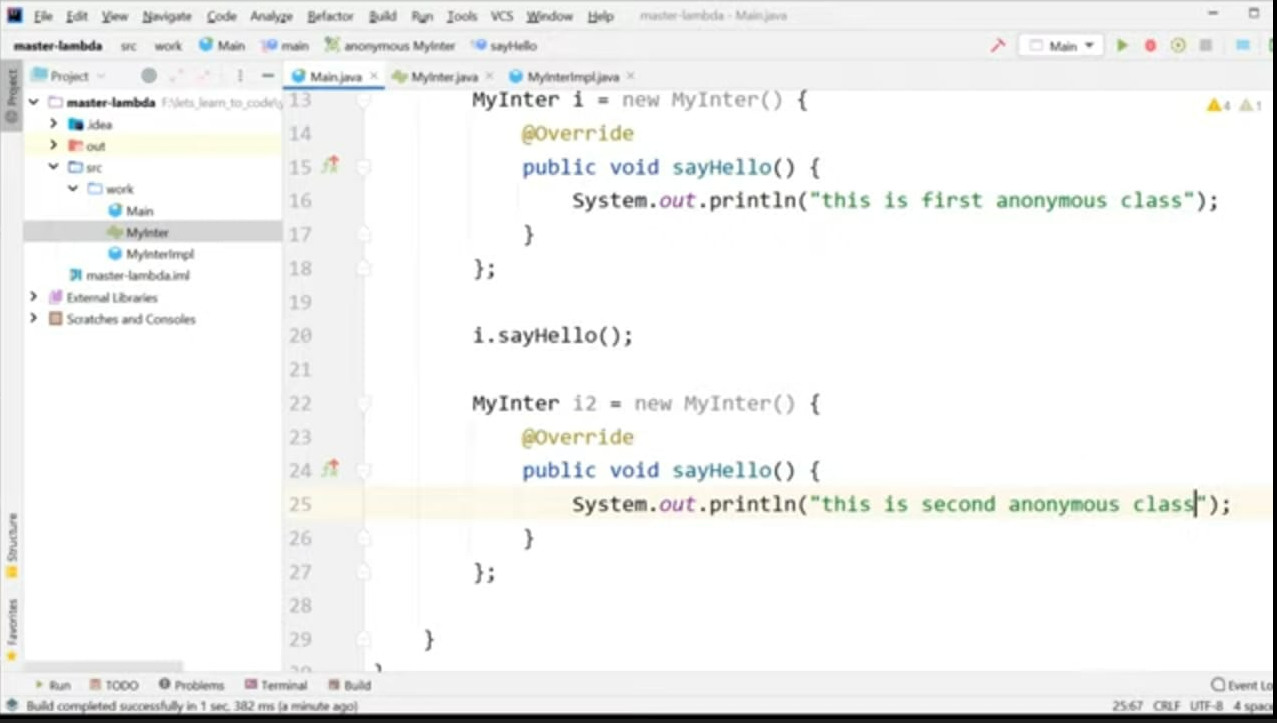


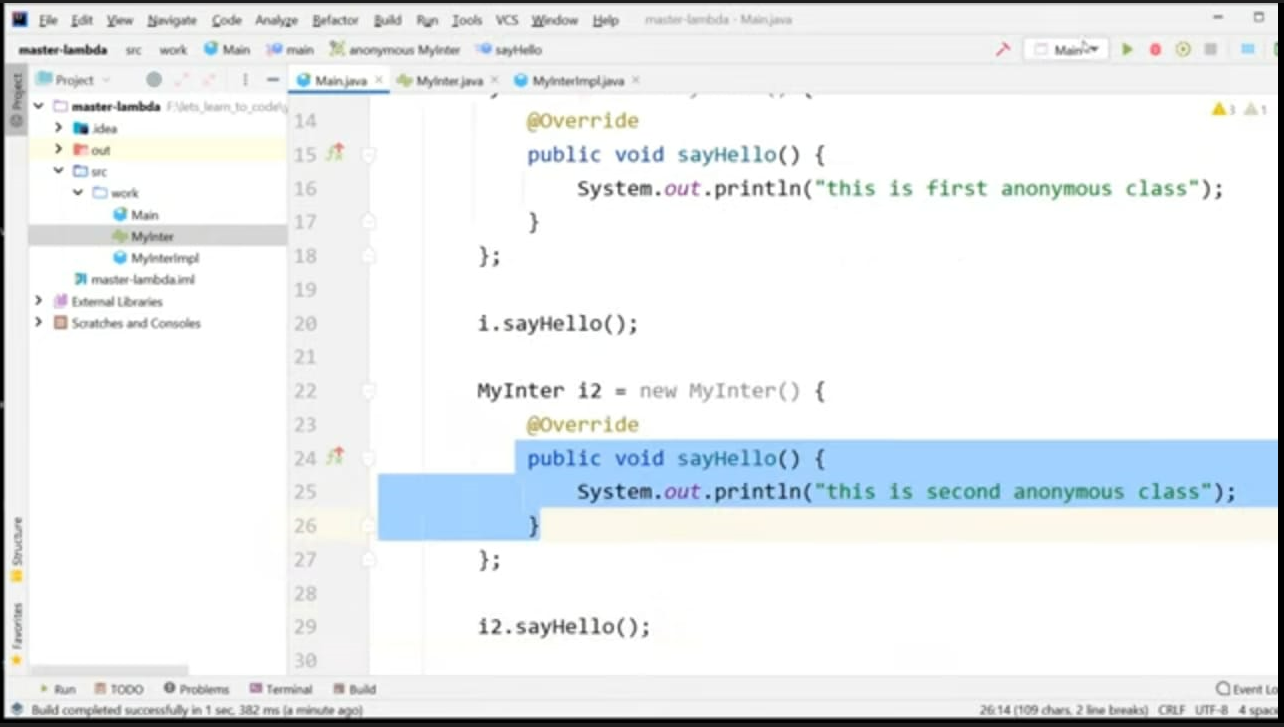


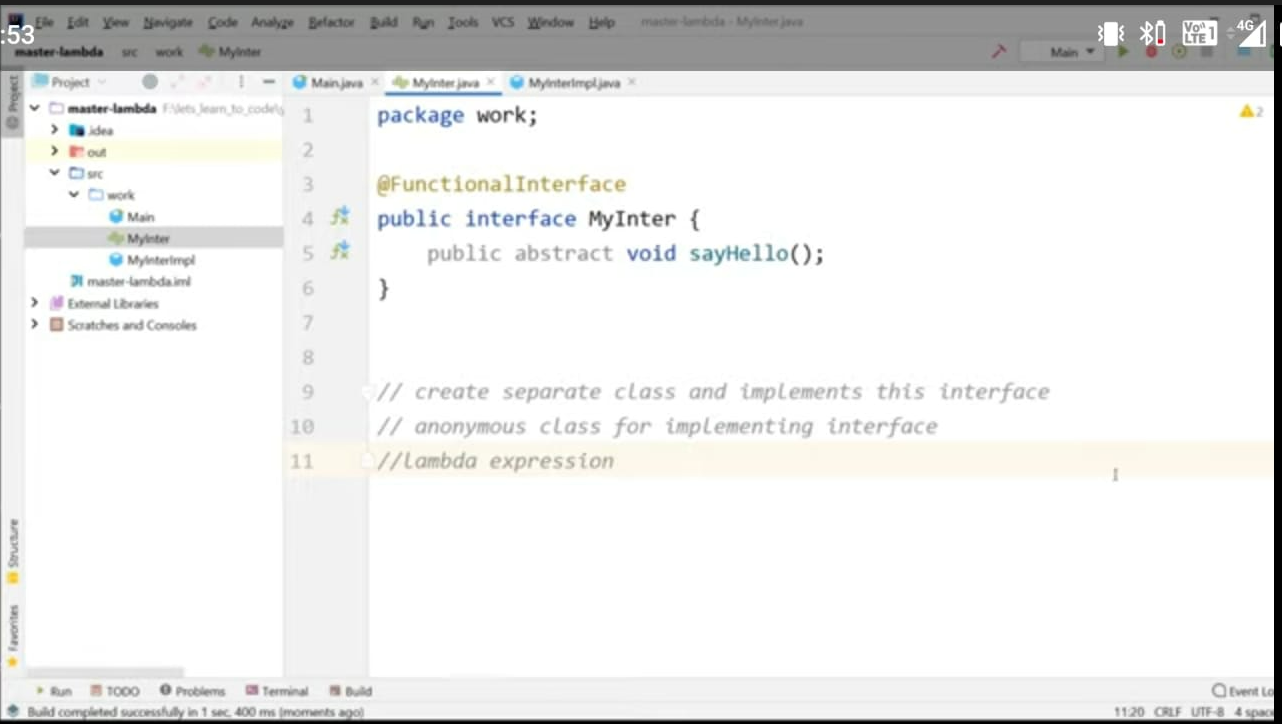
We can also implement interface using anonymous class.



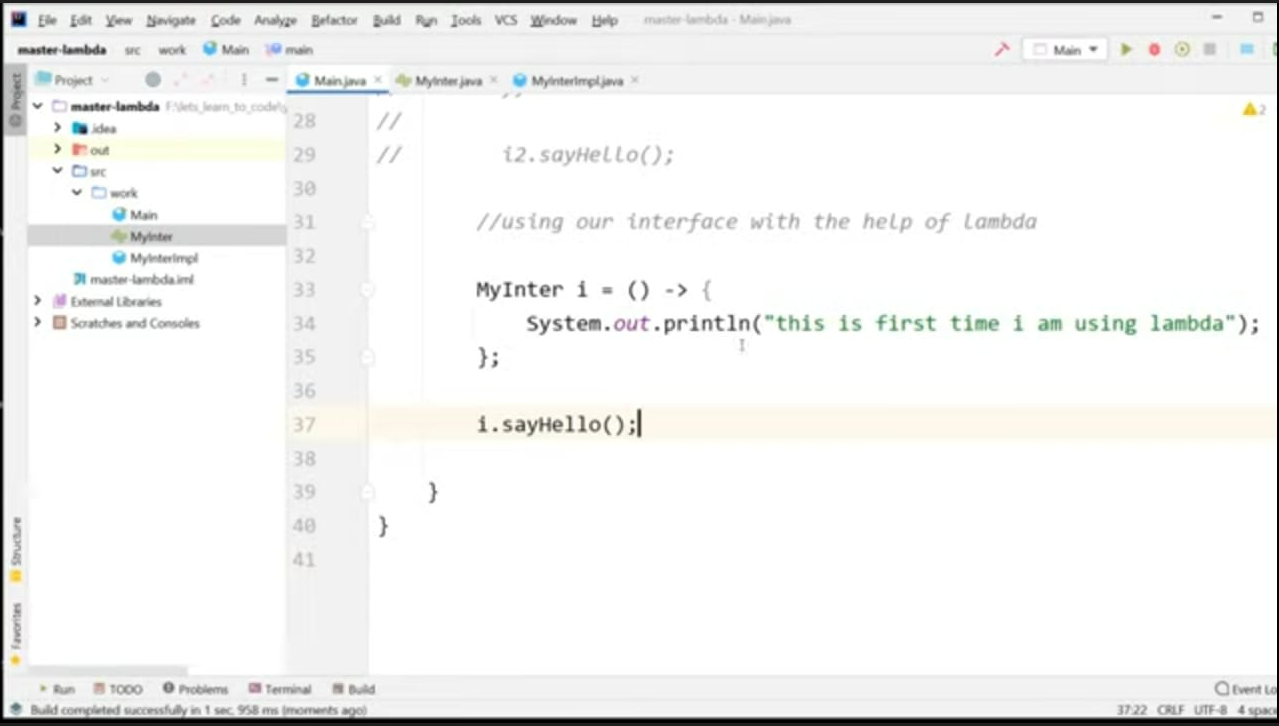




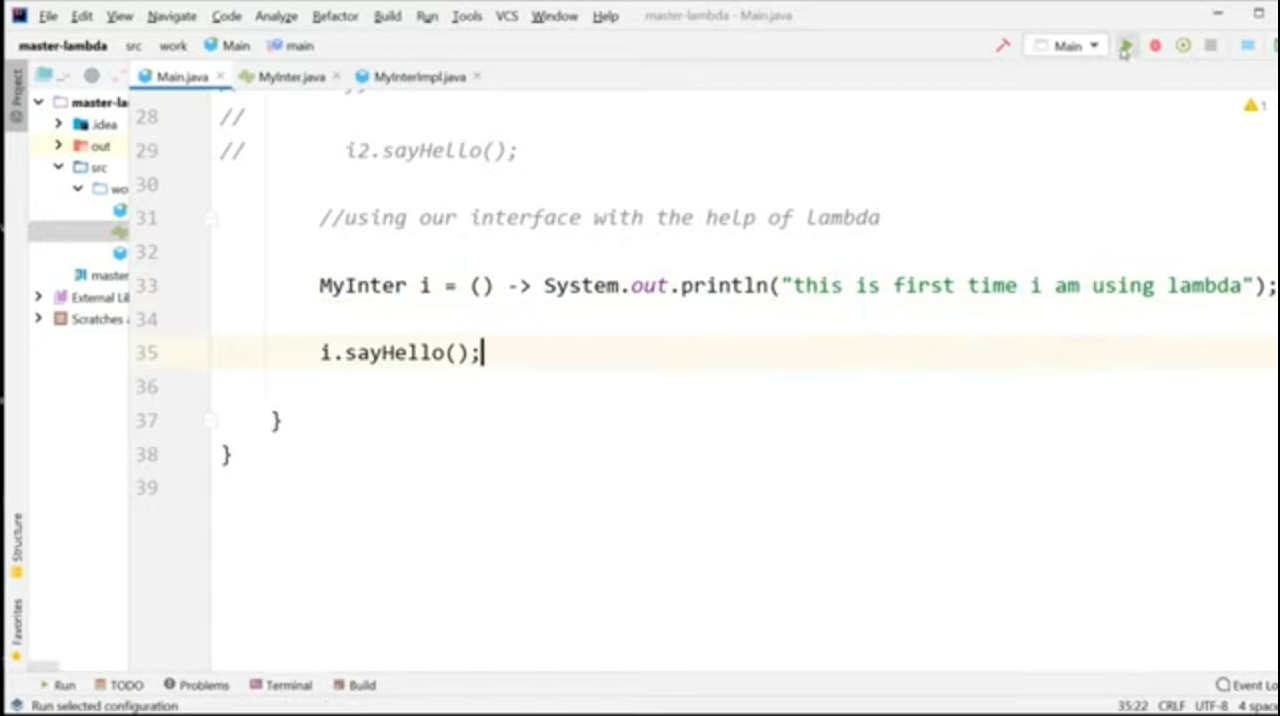




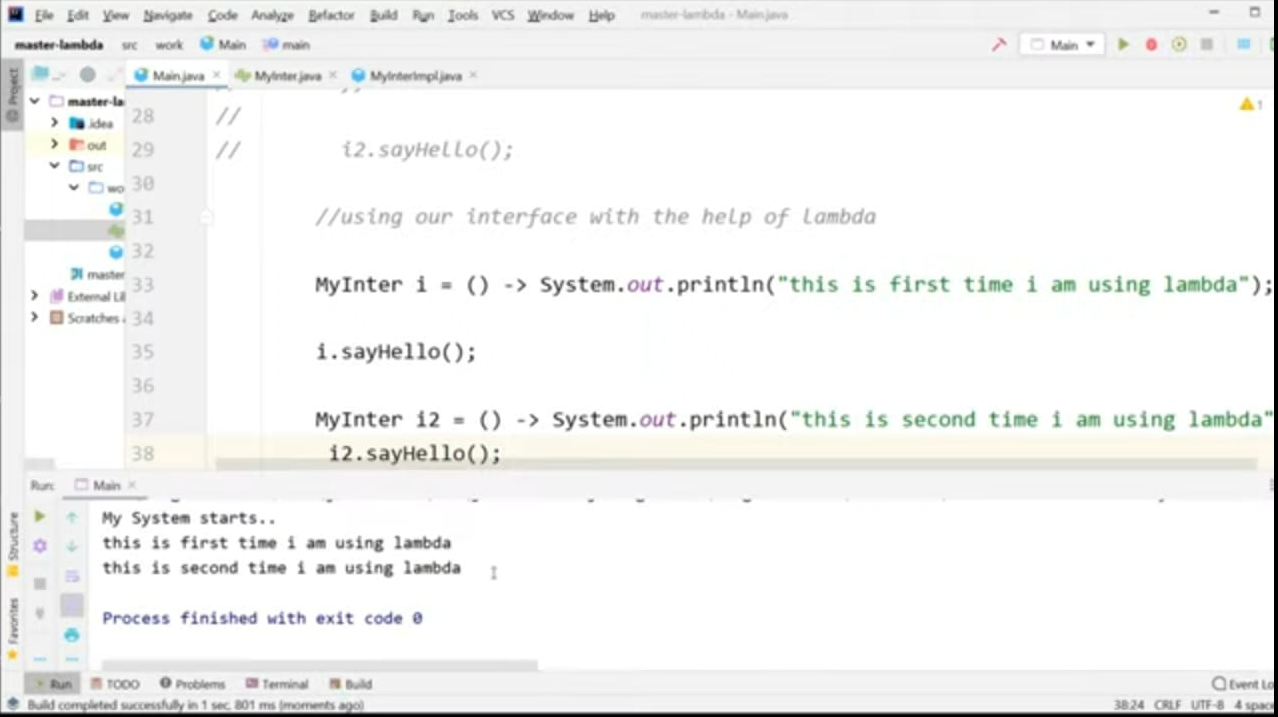
Using Lambda expression, it’s very easy to implement functional interface.

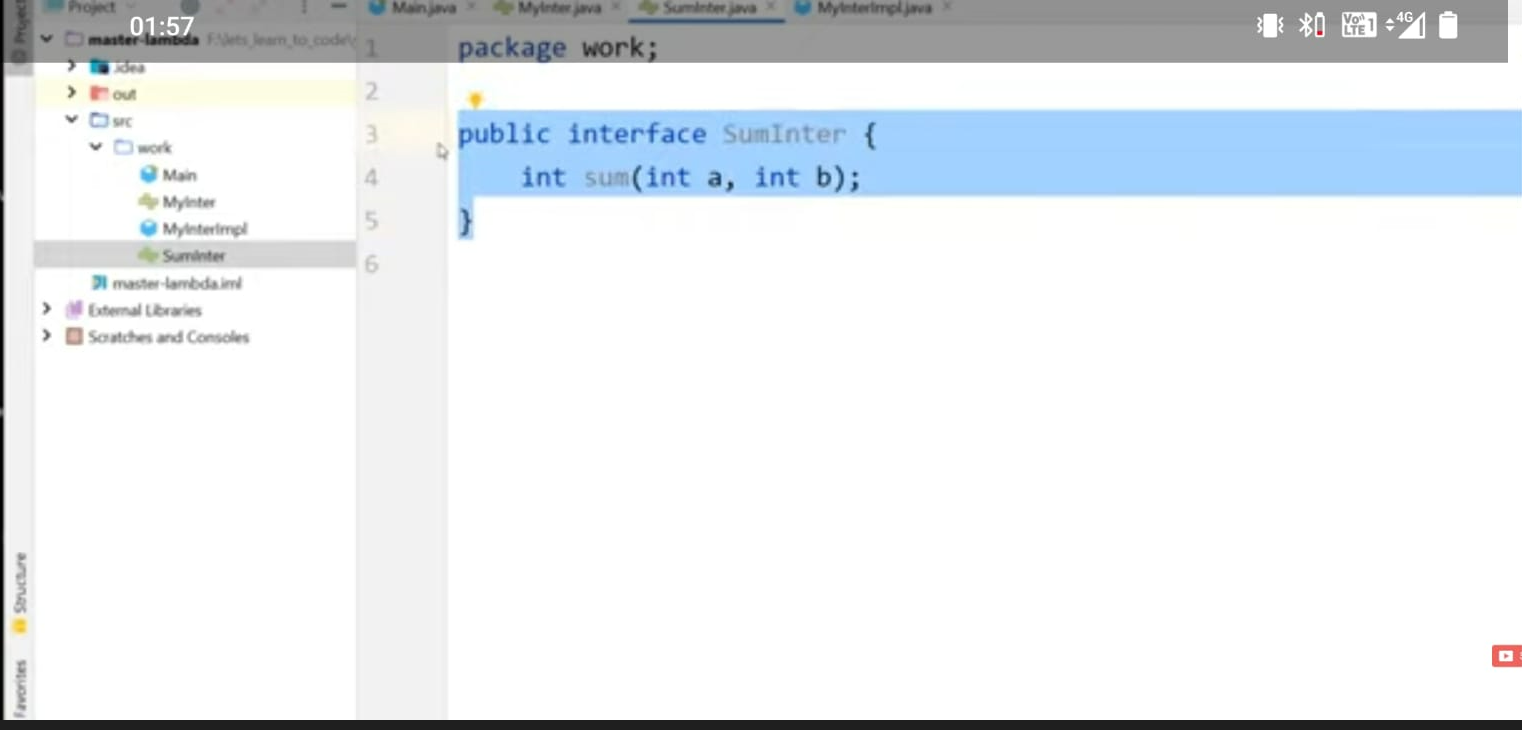


Here using Lambda expression.

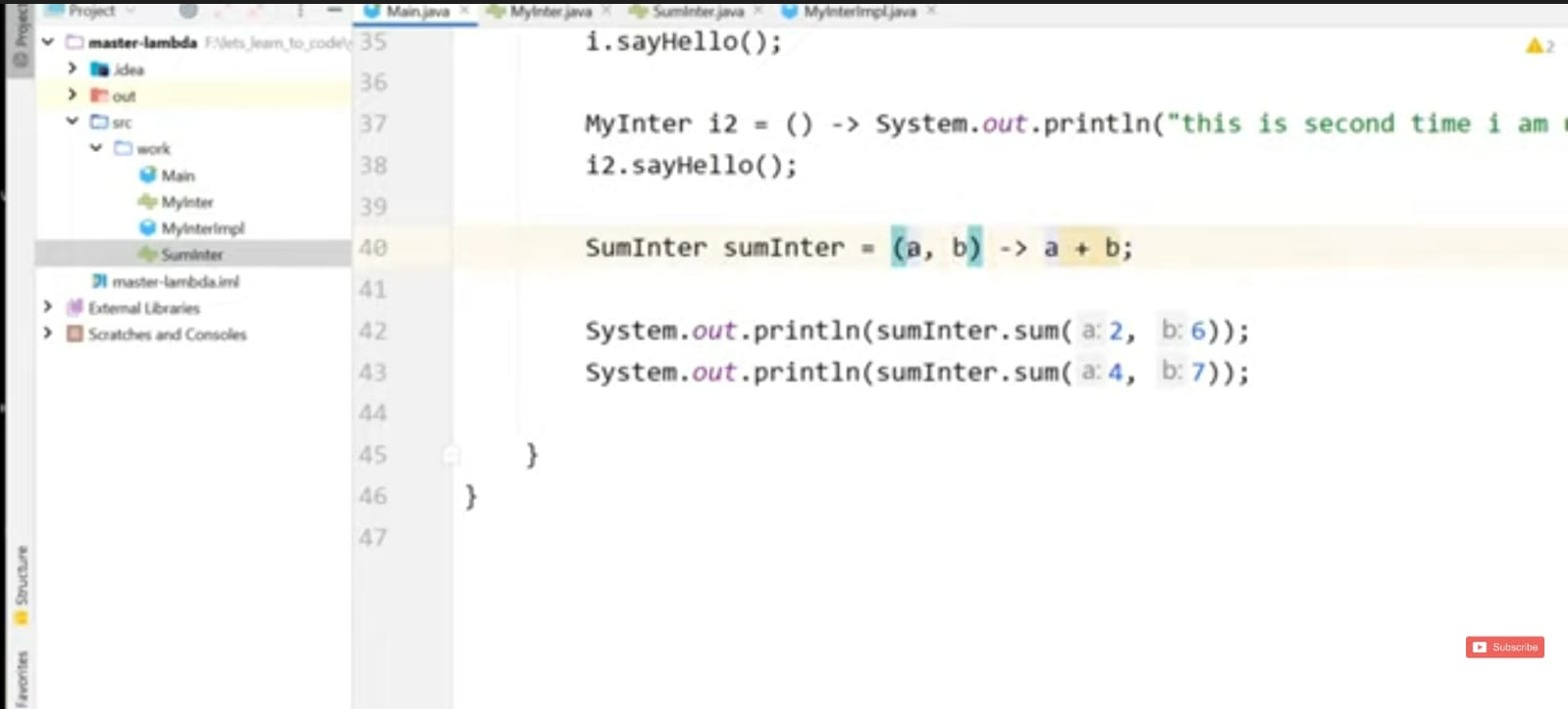


We can eliminate curly braces as it is having only one statement.

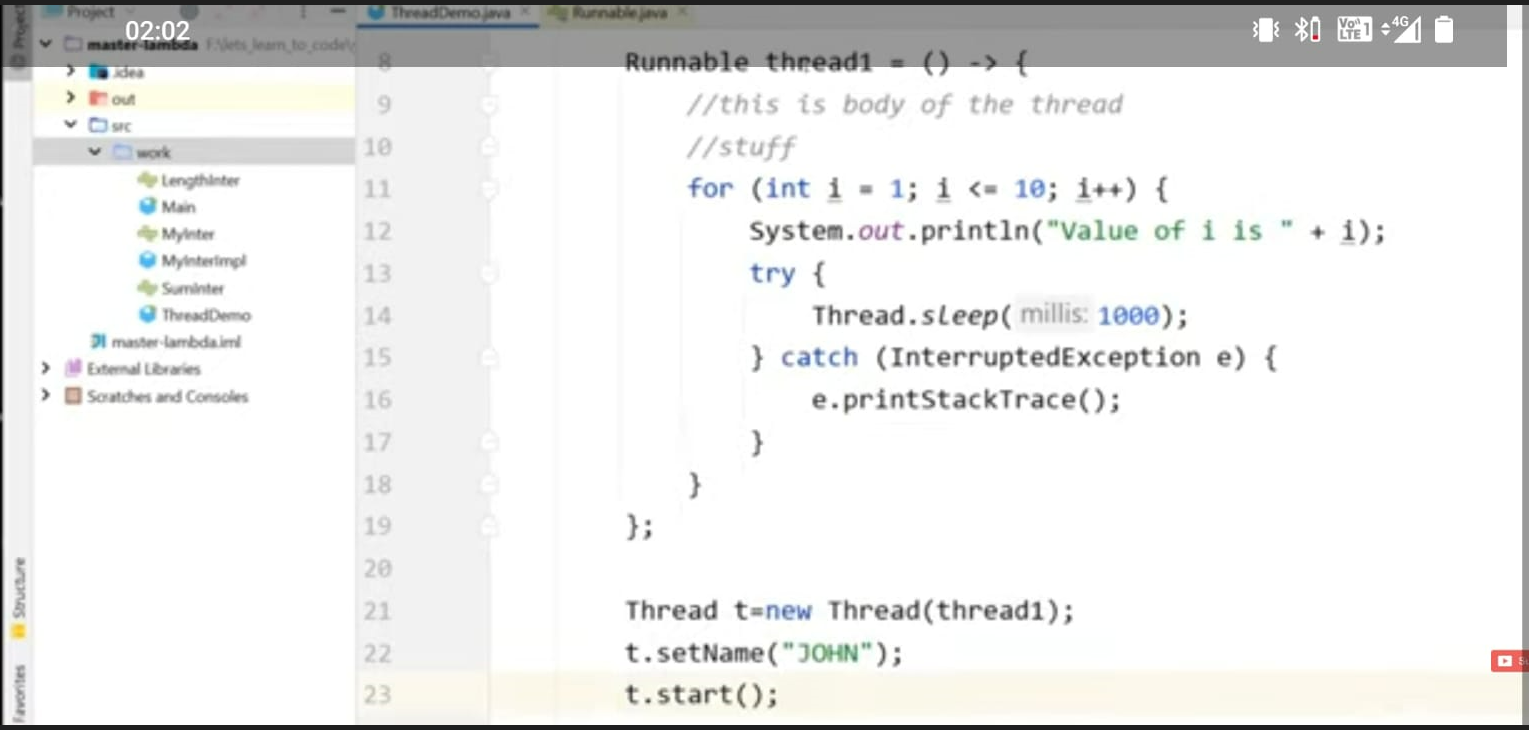








We can eliminate the type as compiler automatically understands from interface method signature and curly braces because it is having only one statement in body.



We can create n number of threads using Lambda expression.

