Functional Interfaces

The interface that has only one abstract method (except from methods of Object) is Functional Interface. Prior to Java 8 they were called Single Abstract Method (SAM) type.

These methods can be overridden with different parameters in super interfaces.

Let us see different Functional Interfaces.

**public** **interface** Runnable {

**public** **abstract** **void** run();

}

Comparator<T> is Functional Interface because it has one abstract non-Object method. equals(Object obj) is Object class method.

**public** **interface** Comparator<T> {

**int** compare(T o1, T o2);

**boolean** equals(Object obj);

}

Following interface is Not Functional because it has Object class method and no method which is not method of Object class.

**public** **interface** NonFunctional {

**boolean** equals(Object obj);

}

Following interface is Functional. It extends NonFunctional interface with one Object class’s method and compare(..) method which is abstract and not in Object class.

**public** **interface** Functional **extends** NonFunctional{

**int** compare(String str1, String str2);

}

You can refer to <https://docs.oracle.com/javase/specs/jls/se8/html/jls-9.html#jls-9.8> for more information.