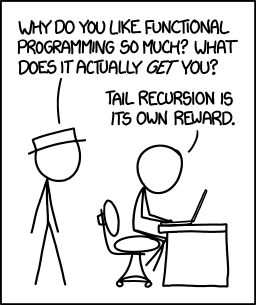
 ****

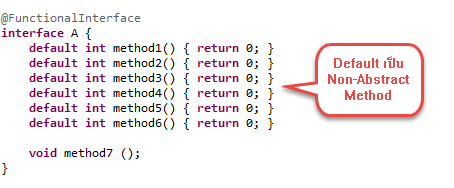
**จุดสำคัญ (Key Points)**

* A functional interface is any interface that has exactly one abstract method.

เป็น Interface ที่มีแค่ 1 Abstract Method เท่านั้น เรียกว่า **“Single Abstract Method (SAM) interfaces”**

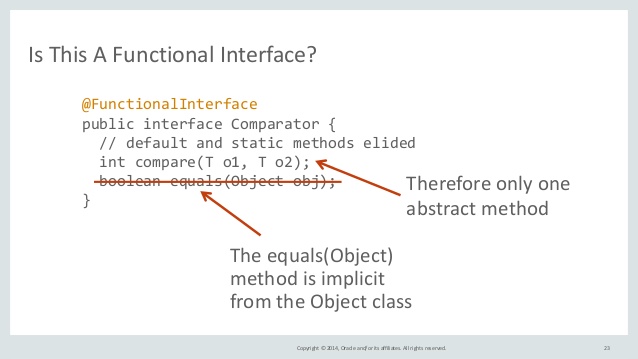
* Since default methods have an implementation, they are not abstract so a functional interface can have any number of them.

นับตั้งแต่นำ Default Methods มาใช้ และมันก็ไม่ได้เป็น Abstract ทำให้เราสามารถประกาศใช้เท่าไหร่ก็ได้

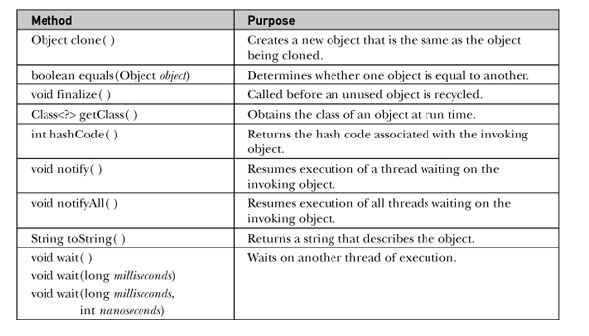


* If an interface declares an abstract method with the signature of one of the methods of java.lang.Object, it doesn't count toward the functional interface method count.

ถ้า interface ประกาศ Abstract Method ที่เป็น Signature ของคลาส Object จะไม่ถูกนับเป็น Abstract Method ในเรื่อง Functional Interface



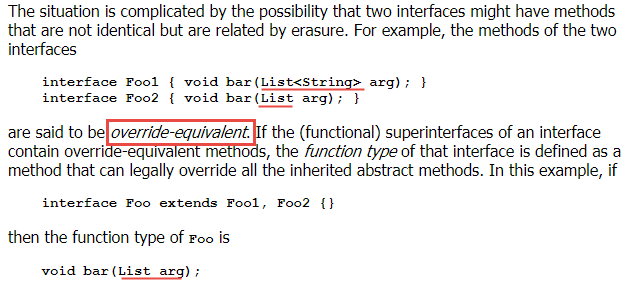
Signature of Class Object Method



**จุดสำคัญ (Key Points)** (ต่อ)

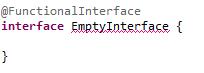
* A functional interface is valid when it inherits a method that is equivalent but not identical to another.

Functional Interface จะใช้ได้ก็ต่อเมื่อมัน inherit ตัว Method ที่เทียบเท่า แต่ไม่เหมือนกัน



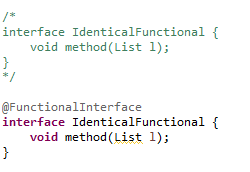
* An empty interface is not considered a functional interface.

Empty Interface จะไม่ถูกเรียกว่าเป็น Functional Interface



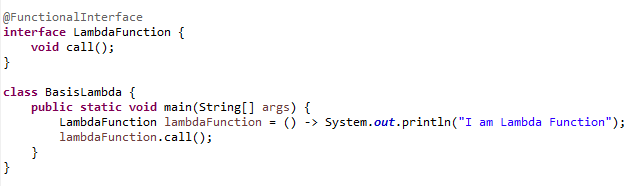
* A functional interface is valid even if the @FunctionalInterface annotation would be omitted.

Functional Interface สามารถใช้ได้ ถึงแม้จะละเว้นประกาศ @FunctionalInterface

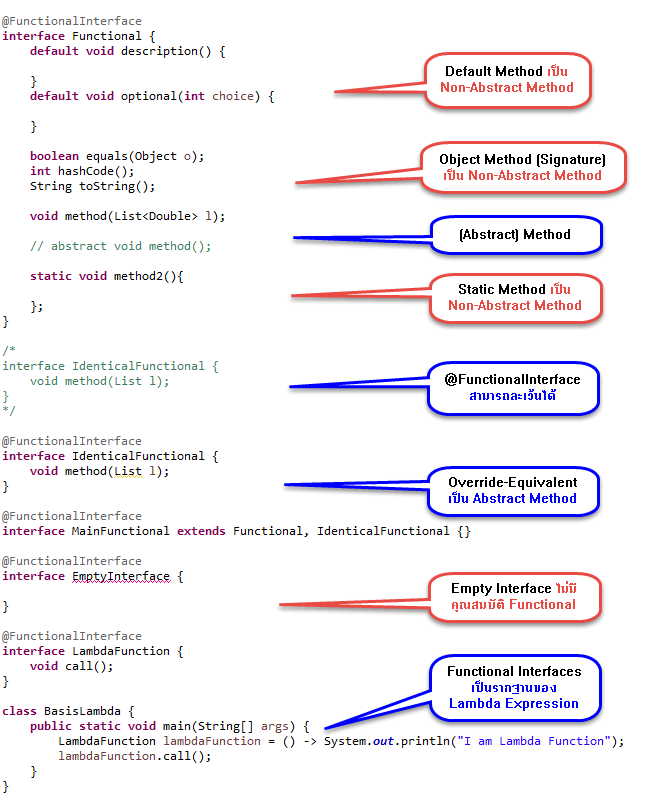


* Functional interfaces are the basis of lambda expressions.

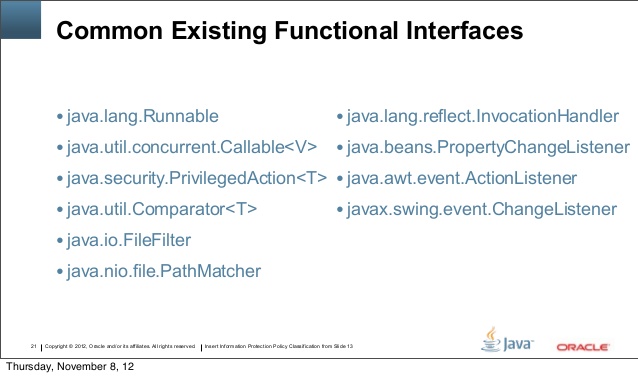
Functional interface ทั้งหมดล้วนเป็นรากฐานของเรื่อง Lambda Expression (บทที่ 9)



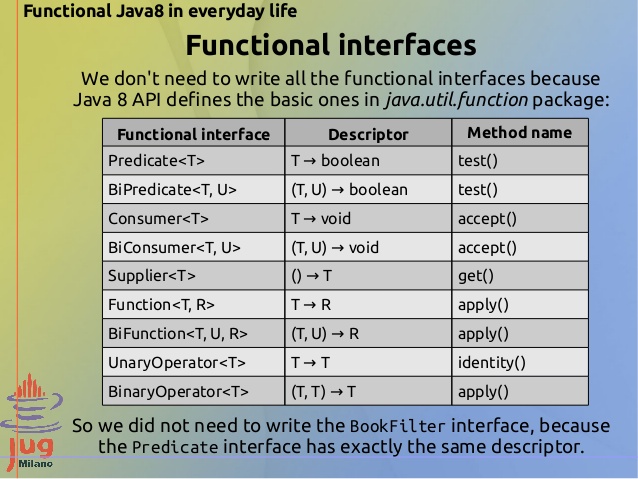
**บทสรุปการประกาศใช้ Functional Interface**



**Common Functional Interfaces in JDK 1-7**



**Newer in JDK 8**

****