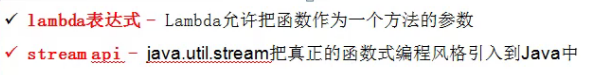
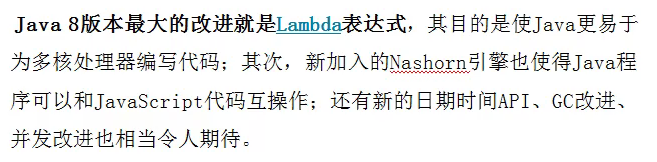
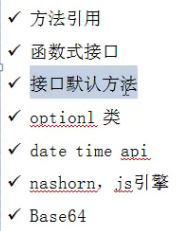
JDK 8的新特性



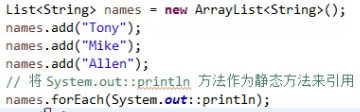
# lambda表达式

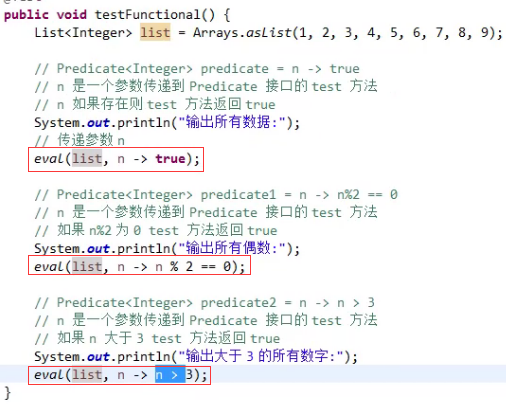
Lambda--允许把函数作为一个方法的参数。



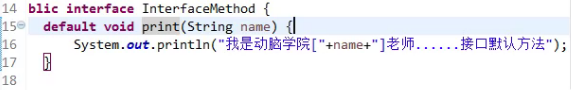


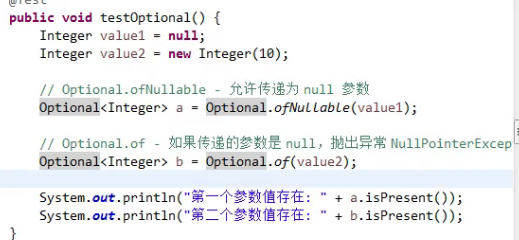
示例：





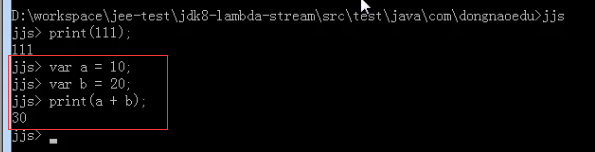
Java接口的方法可以默认实现

  
option类：



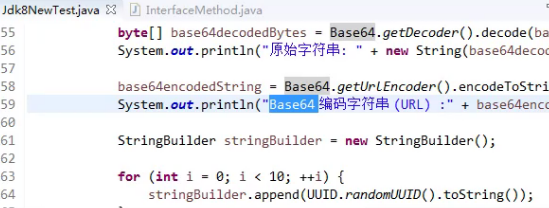
LocalDateTime:



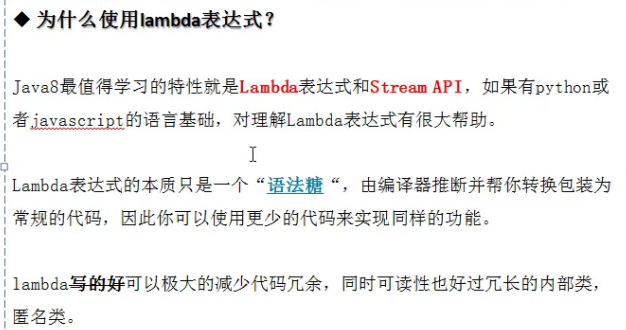


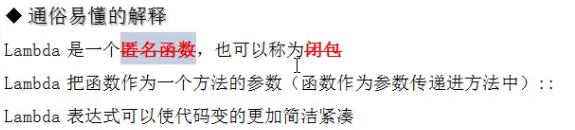


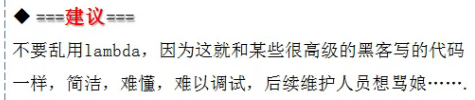
Base64的引入：



## 为什么使用lambda表达式？

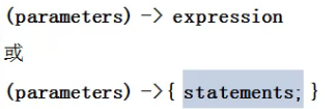






## lambda的语法

两种方式：箭头操作符左边是参数params;右边是Java语句。

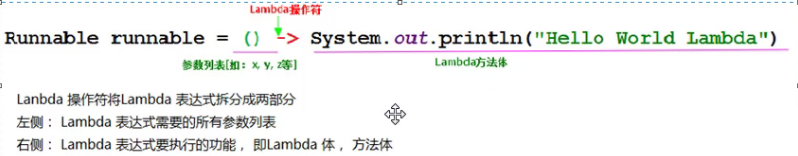


参数列表 -> 函数体；

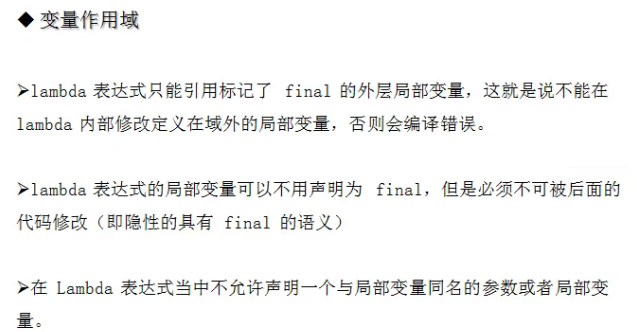
函数体多于一条语句的可用大括号括起。

“Lambda 表达式”(lambda expression)是一个匿名函数，Lambda表达式基于数学中的λ演算得名，直接对应于其中的lambda抽象(lambda abstraction)，是一个匿名函数，即没有函数名的函数。Lambda表达式可以表示闭包。





## lambda的变量作用域



双冒号的作用： (A::B)表示A的B方法。

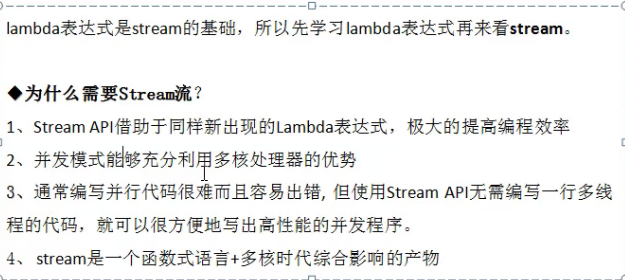
# stream api – java-util.stream

把真正的函数式编程风格引入到Java中。

**Java 8 中的Stream是对集合（Collection）对象功能的增强。**

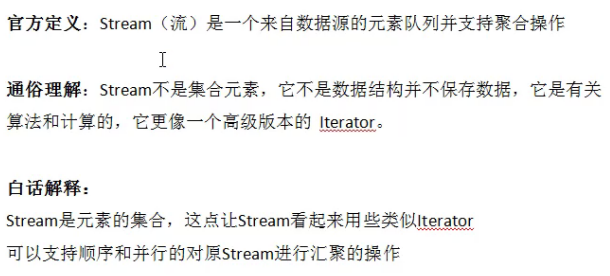
**多核并行执行，执行效率高**

## 为什么需要Stream流？

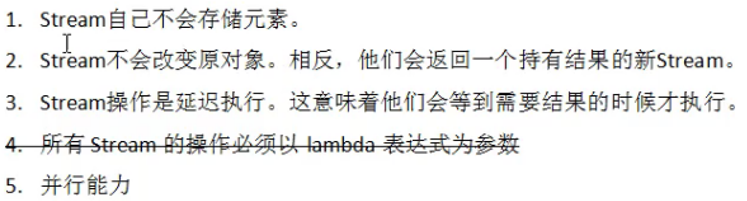


## 什么是Stream流？

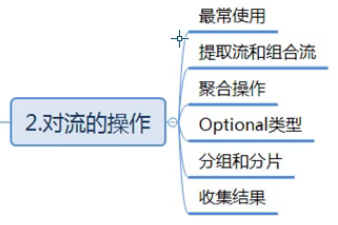
**对集合的增强**。

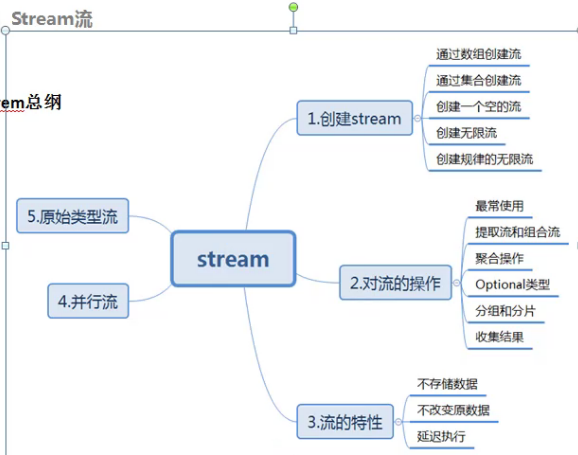


## Stream流的特性

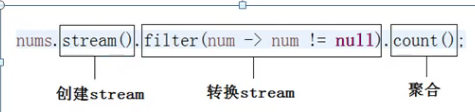


**重点：**

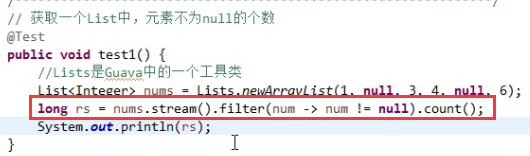




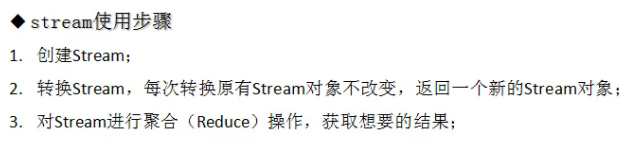
## Stream流的语法



filter()返回的也是stream对象。



## Stream的使用步骤



## Stream的实战

