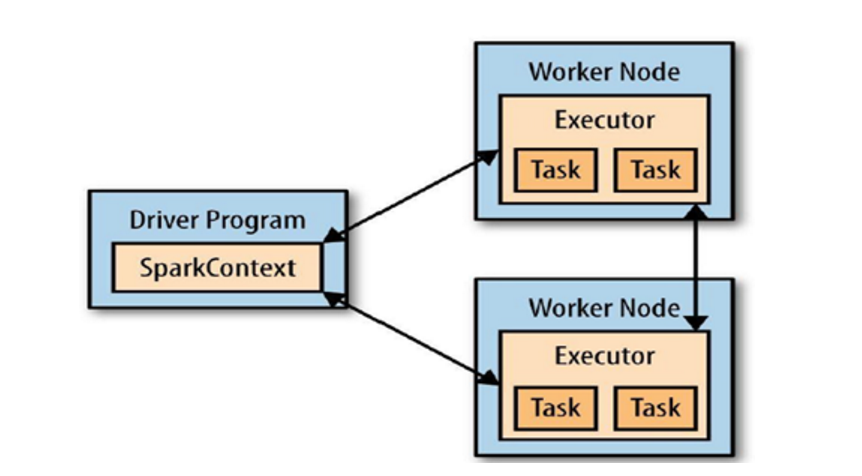
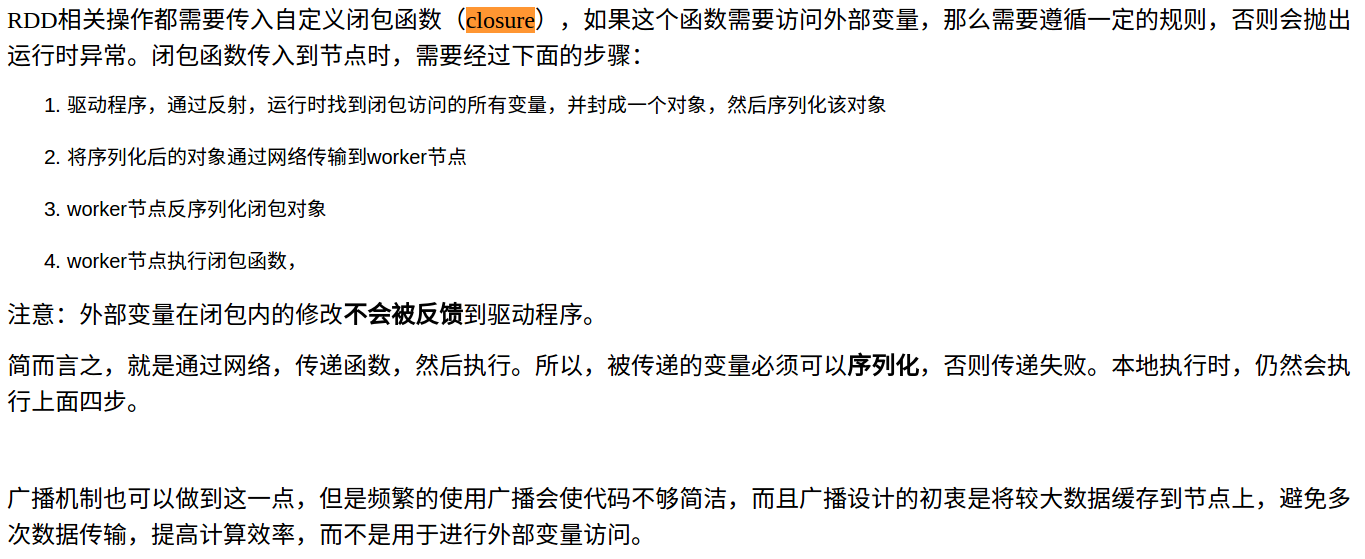
Spark Closure

1:spark 中闭包的理解

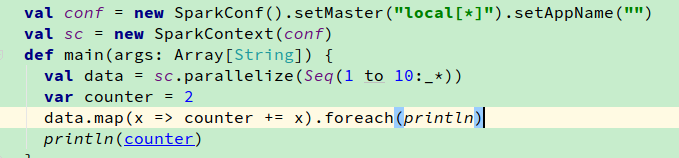
闭包的大致作用就是：函数可以访问函数外面的变量，但是函数内对变量的修改，在函数外是不可见的。BTW, watch :\_\*





**附录中有来自stackOverflow的一个例子**

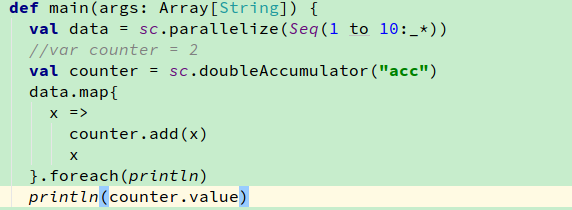
Ok,实验如下：



实验结果：可以看到闭包的存在，map相当于是一个闭包，counter是没有累加器的效果。



正确的程序如下：



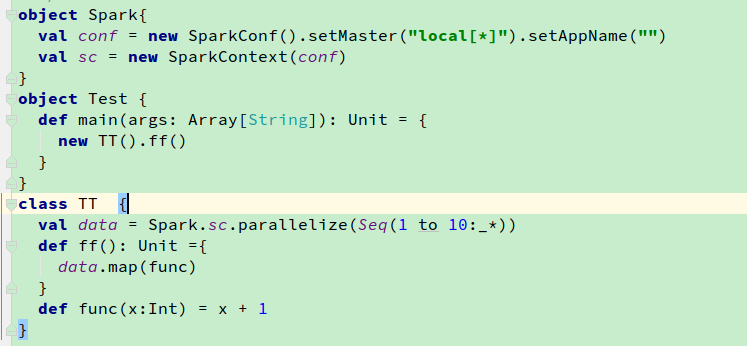
实验结果



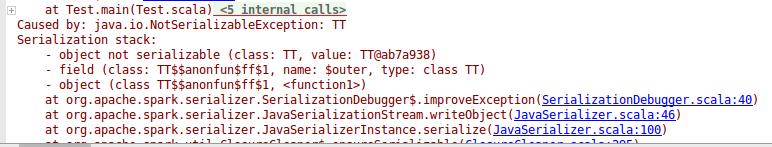
**附录**

**<http://stackoverflow.com/questions/22592811/task-not-serializable-java-io-notserializableexception-when-calling-function-ou>**

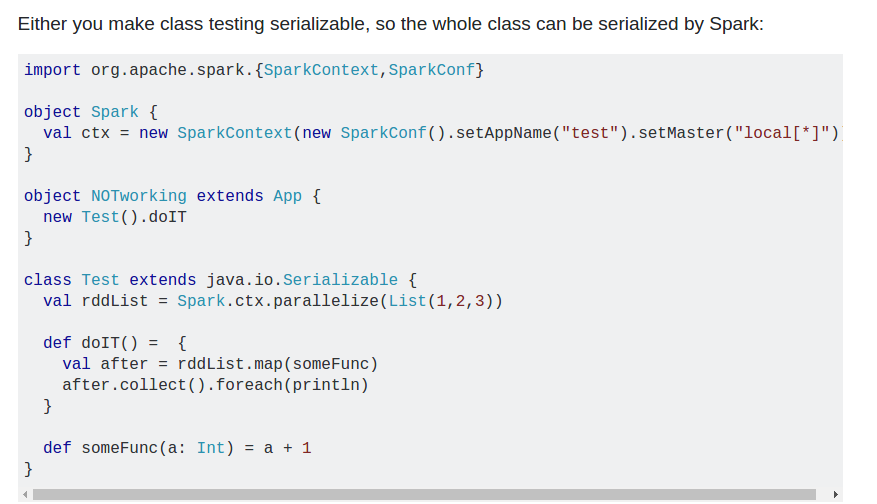
**Error:Task not serializable: java.io.NotSerializableException when calling function outside closure only on classes not objects**



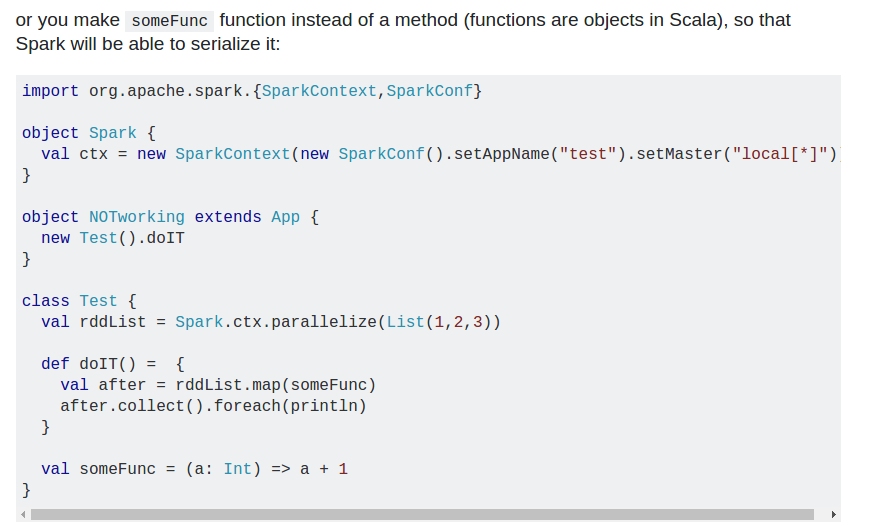
Error:



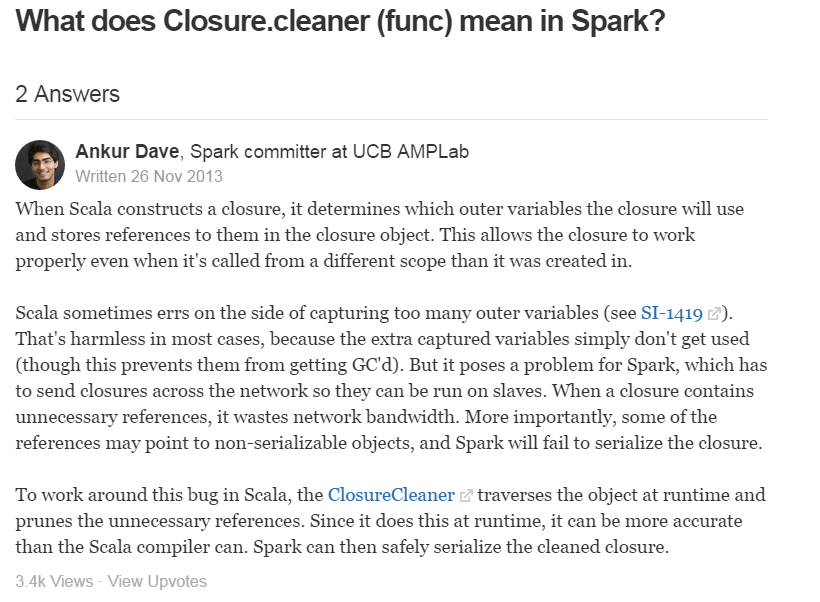
解决思路：



While methods can not be serialized by them own



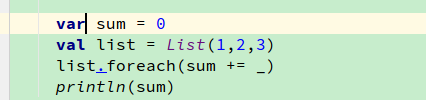
**在spark 中于是就有了clean closure的操作：**

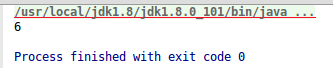


具体可以参考spark core-SparkContext 中关于闭包的部分。

2:Scala 中闭包的理解

实验：似乎这里的闭包与Spark中的概念不一样





类似功能可以使用java 内部类实现，并且java8 的lambda 计划已经加入了函数闭包