Spark程序编写：继承App的问题

我们知道，编写Scala程序的时候可以使用下面两种方法之一：

```
object IteblogTest extends App {
  //ToDo
}

object IteblogTest{
  def main(args: Array[String]): Unit = {
    //ToDo
  }
}
```

上面的两种方法都可以运行程序，但是在Spark中，第一种方法有时可不会正确的运行（出现异常或者是数据不见了）。比如下面的代码运行的时候就会出现异常：

```
object IteblogTest extends App{
  val conf = new SparkConf().setAppName("Iteblog")
  val sc = new SparkContext(conf)
  val sample = sc.parallelize(1 to 100)
  val bro = sc.broadcast(0)
  val broSample = sample.map(x => x.toString + bro.value)
  broSample.collect().foreach(println)
}
```

运行上面的程序会出现以下的异常：

```
15/12/10 17:14:49 WARN scheduler.TaskSetManager: Lost task 0.0 in stage 0.0 (TID 0, www.iteblog.com): java.lang.NullPointerException
  at com.iteblog.demo.IteblogTest$$anonfun$1.apply(IteblogTest.scala:13)
  at com.iteblog.demo.IteblogTest$$anonfun$1.apply(IteblogTest.scala:13)
  at scala.collection.Iterator$$anon$11.next(Iterator.scala:328)
  at scala.collection.Iterator$class.foreach(Iterator.scala:727)
  at scala.collection.AbstractIterator.foreach(Iterator.scala:1157)
  at scala.collection.generic.Growable$class.$plus$plus$eq(Growable.scala:48)
  at scala.collection.mutable.ArrayBuffer.$plus$plus$eq(ArrayBuffer.scala:103)
```

```
at scala.collection.mutable.ArrayBuffer.$plus$plus$eq(ArrayBuffer.scala:47)
at scala.collection.TraversableOnce$class.to(TraversableOnce.scala:273)
at scala.collection.AbstractIterator.to(Iterator.scala:1157)
at scala.collection.TraversableOnce$class.toBuffer(TraversableOnce.scala:265)
at scala.collection.AbstractIterator.toBuffer(Iterator.scala:1157)
at scala.collection.TraversableOnce$class.toArray(TraversableOnce.scala:252)
at scala.collection.AbstractIterator.toArray(Iterator.scala:1157)
at org.apache.spark.rdd.RDD$$anonfun$collect$1$$anonfun$12.apply(RDD.scala:909)
at org.apache.spark.rdd.RDD$$anonfun$collect$1$$anonfun$12.apply(RDD.scala:909)
at org.apache.spark.SparkContext$$anonfun$runJob$5.apply(SparkContext.scala:1850)
at org.apache.spark.SparkContext$$anonfun$runJob$5.apply(SparkContext.scala:1850)
at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:66)
at org.apache.spark.scheduler.Task.run(Task.scala:88)
at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:214)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
at java.lang.Thread.run(Thread.java:744)
```

这是因为：如果你继承了App trait，那么里面的变量被当作了单例类的field了；而如果是main方法，则当作是局部变量了。而且trait App是继承了DelayedInit，所以里面的变量只有用到了main方法的时候才会被初始化。

> It should be noted that this trait is implemented using the [[DelayedInit]] functionality, which means that fields of the object will not have been initialized before the main method has been executed.

为了程序能够正常地运行，最好不要继承App，直接用main方法。甚至在Spark官方文档也提出最好不要用App

> Note that applications should define a `main()` method instead of extending `scala.App`.Subclasses of `scala.App` may not work correctly.

不过如果你实在是想继承App trait，可以将代码写成下面的格式：

```
object IteblogTest extends App {{
  //ToDo
}}
```

本文链接: 【】（）