

we write about the things we build and the things we consume

search metabroadcast

written by [Max Tomassi](#) on 5 November 2014 in [engineering](#),
[real-time data](#)



stop your spark streaming application gracefully

Update, 14 January 2016: See comment below from Matt; this approach no longer works in newer versions of Spark.

You've just submitted your [Spark Streaming](#) application to your cluster and all your trustworthy workers and receivers are diligently doing their precious jobs. Receivers are consuming messages from your data source (e.g. Kafka) and notifying the driver program. The driver program will then schedule tasks to the workers in order to have those messages processed (e.g. transformed and stored somewhere). Now what if you need to restart or redeploy your application?

stop gracefully or you lose data

Yes, if you just stop the application, for example killing the driver process on your master node, you're not guaranteed that all the messages will be processed. This is because your receivers might have already consumed messages that your workers haven't processed yet. When you stop the application, the driver will just shutdown discarding the tasks that still need to be performed. So how can we avoid dataloss? The [Spark Streaming programming guide](#) suggest to invoke the method `StreamingContext.stop(...)`, if you are in the Scala world, or `JavaStreamingContext.stop(...)`, if you are in the Java one. Doing this will "ensure data that have been received is completely processed before shutdown"

ok, but how

sign up to #metabeers

the monthly meeting of minds & metadata over pints & treats in London. and a fine publication, too

talks + wildcard guests + the only metadata pub quiz with blooming prizes + fast games of quoridor

email address

metabroadcasts require

1 creative technologist | 1 designer at large | 1 project manager | 7 software engineers | 2 systems engineers | 1 team lead

blog



[is vr finally going mainstream?](#)



[atlas release notes - week to 12th october 2016](#)



[the ultimate insults](#)



[engineer fuel \(and how to make it\)](#)



[protocol buffers](#)

slideshow

A Spark Streaming application is a long running application, so it's not so evident from where you can call that method when you decide it's time to shut it down. I [asked](#) for more detail in the Spark mailing list, receiving the suggestion to implement an HTTP service exposed by the Spark application that will trigger the `StreamingContext.stop(...)` (we're writing in Scala) method. It actually makes sense, but is it worth to spawn an HTTP server just to be able to stop the application? There should be an easier way.

Thinking about it, the easiest solution would be to handle the SIGTERM signal that is sent to the driver process when you kill it. With Scala this can be easily achieved using a shutdown hook and specifically `sys.ShutdownHookThread`. The following is a snippet of code that shows how to register a shutdown hook.

```
1 def main(args: Array[String]) {
2
3   // Prepare your environment
4
5   val ssc = new StreamingContext(conf, Seconds(batchDurationInSec))
6
7   // Do your processing
8
9   sys.ShutdownHookThread {
10     log.info("Gracefully stopping Spark Streaming Application")
11     ssc.stop(true, true)
12     log.info("Application stopped")
13   }
14
15   ssc.start
16   ssc.awaitTermination
17
18 }
```

StopSparkStreaming.scala hosted with ❤ by GitHub

[view raw](#)


As you can see in the code I'm passing two booleans to the `stop` method. As the [documentation](#) suggest, the first one is to stop the Spark context and the second one is to stop the streaming application gracefully. Obviously, a similar thing can be done in Java using [Runtime.addShutdownHook\(...\)](#)

If you now look at the logs when you kill your driver process, you'll see that before actually stopping the application a number of tasks will still be scheduled and performed by the executors. So you've now guaranteed that all the consumed messages will actually be processed.

hope it helps!


This was a small tip, but I hope it can be useful for who doesn't want to lose data when stopping a Spark streaming application. If you found a different and better way to achieve the same result, please let us know with a comment below or [tweeting us](#).

Tweets by @MetaBroadcast

**MetaBroadcast**
@MetaBroadcast

Atlas release notes - week to 12th October 2016
[metabroadcast.com/blog/atlas-rel...](#) ^AdM

13 Oct

**MetaBroadcast**
@MetaBroadcast

How about a bit of D&D?
[metabroadcast.com/blog/the-ultim...](#) ^LH

13 Oct

[Embed](#)

[View on Twitter](#)



free whitepaper

let's work together

[slideshow](#)