# Exception using BlobWritePartial and FpFileInputStream

newtopic locked SDK

View previous topic :: View next topic

| Author | Message |
|---|---|
| **stefano.valsecchi@txt.it**<br>Newbie<br>  <br>Joined: Nov 14, 2007<br>Posts: 7 | Posted: Wed Mar 05, 2008 10:09 pm Post subject: Exception using BlobWritePartial and FpFileInputStream quote<br><br>Good morning,<br><br>We are implementing a routine whose target is to archive a single, large file in Centera storage and we have found some problem with blob slicing technique:<br><br>Here is a portion of our code (variable are replaced with values for clarity):<br><br>//Open the second slice of a file. Slice size is 100.000 bytes. File size is 9 Mbyte<br>FPFileInputStream fis = new FPFileInputStream(new File("c:/test.mpeg"), 100000, 100000);<br><br>// Slice writing<br>myclip.BlobWritePartial(fis, FP_OPTION_SERVER_CALCID_STREAMING, 2);<br><br>If slice size is greater than internal buffer size (16K), the result is an FPLibraryException wrapping the following IndexOutOfBoundException<br><br>java.lang.IndexOutOfBoundsException |

at java.io.RandomAccessFile.readBytes(Native Method)
at java.io.RandomAccessFile.read(RandomAccessFile.java:307)
at com.filepool.fplibrary.FPFileInputStream.read(Unknown Source)
at it.txt.polymedia.feeds.actions.video.CenteraArchiver $MonitoredFileInputStream.read(CenteraArchiver.java:845)
at com.filepool.natives.FPLibraryNative.FPTag_BlobWritePartial (Native Method)
at com.filepool.fplibrary.FPTag.BlobWritePartial(Unknown Source)

Inserting a breakpoint in method read of class RandomAccessFile, we noted a call with the following parameters:

RandomAccessFile.read(buffer, 100000, 16384);

Where buffer is a byte array of 16384 elements (the internal buffer, I suppose)
Our suspect is that FPFileInputStream class store the offset received in the constructor and passes it to read method (Other test with different offset values confirm this suspect).
The problem is that the offset parameter of read method refer to the buffer and not to the file!!!

Is it a bug of FPFileInputStream class or there is something wrong in our code? Can somebody help?

Thanks

Bye

P.S.: I have found another topic on this forum for the same problem with title blobwritepartial problem and last post date 16/03/2007, but it doesn't solved.

Back to top

**mikehorgan**
Power user

Joined: Mar 25, 2005
Posts: 483
Location: Hopkinton, MA, USA

Posted: Sat Mar 08, 2008 12:44 am Post subject:

Hello Stefano -

First off, thank you for your SDK bug report. I have successfully confirmed this issue and submitted it to our SDK engineering team. A fix will be included in the next scheduled release of our SDK (version 3.2 SP1 scheduled for later this year, exact date TBD).

In order to work around this SDK issue, you will need to implement your own 'partial' file stream by extending a java.io. InputStream subclass with an implementation of the com. filepool.fplibrary.FPStreamInterface interface.

Be aware that if you decide to implement the FPMark/FPReset methods in this interface you will need to test these thoroughly, as an incorrect implementation of these methods can cause unrecoverable data loss whenever your program encounters network or Centera conditions that require the SDK to rewind and replay your stream's data.

Also, we have recently discovered a software flaw in the 3.2.607 SDK which affects multithreaded blob slicing clients such as yours. We will be releasing a patched 3.2 SDK (version 3.2 patch 1) before the end of this month; however I am sorry to report that it is too late in the process to include a fix for your FPFileInputStream issue in this pending release.

Given these constraints, you should develop and test your partial strem code using SDK 3.1 SP1 (3.1.544) but plan to adopt 3.2p1 prior to your final code release. When 3.2 SP1 becomes available later this year, I would recommend you back out your own partial file stream implementation and utilize the fixed 'official' version of this feature as implemented by the FPFileInputStream 'offset' constructor.

One last point about blob slicing: We would not recommend that you slice your blobs into smaller than 5MB chunks, and would prefer 20MB or greater for optimal ingest performance with Centera. Slicing content into small chunks as in your example code would be self-defeating from a performance perspective.

Best Regards,
Mike Horgan

_____

Mike Horgan
EMC Centera Integration Architect