Exiv2 - Feature #1232

Support JPEGs encoding and decoding in which the Exif data exceeds 64k bytes

26 Sep 2016 19:38 - Robin Mills

Status:	New	**Start date:**	26 Sep 2016
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	metadata	**Estimated time:**	0.00 hour
Target version:	0.28		

Description

http://dev.exiv2.org/issues/1221#note-8

History

#1 - 26 Sep 2016 19:44 - Robin Mills

- File XmpPart3-2016-page13.png added

Adobe published a new Edition of the XMPsdk on 30 July 2016. The latest version of this document XMP-Toolkit-SDK-CC201607/docs/XMPSpecificationPart3.pdf contains this statement on page 13. This statement was not in the 2010 Edition of that document.

> The JPEG standard does not prescribe ordering among APPn segments, but some related standards do. For example, Exif requires that Exif APP1 segment be immediately after the SOI. Also, some applications improperly assume that the segments are in a particular order. For compatibility, it is best to put the Exif APP1 first, the XMP APP1 next, the PSIR APP13 next, followed by all other marker segments.
>
> ***NOTE*** If the size of the Exif APP1 marker or the PSIR APP13 marker exceeds 64KB, the marker is split into multiple blocks of 64KB size each.

XmpPart3-2016-page13.png
If this is true, we can indeed store huge blobs of Exif metadata. If the metadata exceeds 64k, we can simply divide it into multiple segments. However, I would want to see a reliable example of such a file. By "reliable", I mean it has a blue-chip pedigree (such as PhotoShop). And I'd like to see a specification which is focused on this matter. The statement in this spec doesn't feel definitive and doesn't address APP1 continuation block signatures.

A typical JPEG structure is as follow:

```
696 rmills@rmillsmbp:~/gnu/exiv2/trunk $ exiv2 -pS test/data/Reagan.jpg
STRUCTURE OF JPEG FILE: test/data/Reagan.jpg
 address | marker       |  length | data
       0 | 0xffd8 SOI
       2 | 0xffe1 APP1  |    5718 | Exif..MM.*.....................     <---- This is the Exif data (Tiff Enc
oded)
    5722 | 0xffed APP13 |    3038 | Photoshop 3.0.8BIM.........Z...
    8762 | 0xffe1 APP1  |    5329 | http://ns.adobe.com/xap/1.0/.<?x
   14093 | 0xffe2 APP2  |     576 | ICC_PROFILE......0ADBE....mntrRG chunk 1/1
   14671 | 0xffee APP14 |      14 | Adobe.d@......
   14687 | 0xffdb DQT   |     132
   14821 | 0xffc0 SOF0  |      17
   14840 | 0xffdd DRI   |       4
   14846 | 0xffc4 DHT   |     418
   15266 | 0xffda SOS
697 rmills@rmillsmbp:~/gnu/exiv2/trunk $
```

I think Adobe are saying that if you follow an APP1 segment with one or more APP1 segments of ***exactly*** 64k bytes you should consider it to be a "segmented" continuation of its predecessor. That seems reasonable to me. However it's quite different from the "chunk" mechanism used to allow ICC profiles to exceed a single APP2 segment. Additionally, APP1 segments begin with a ***signature*** such as "Exif\0\0" to identify them to readers.

I've hunted through Phil (of ExifTool fames) repository of 7000 JPG files. Not one has Exif data encoded in this way.

AGFA have files with multiple segments of length 65535. However they are not APP1 segment. Here's the evidence:

```
702 rmills@rmillsmbp:~/gnu/exiv2/trunk $ exiv2 -pS /mmHD/Users/rmills/Jenkins/testfiles/Phils/Agfa/AgfaOPTIMA1
338mT.jpg
STRUCTURE OF JPEG FILE: /mmHD/Users/rmills/Jenkins/testfiles/Phils/Agfa/AgfaOPTIMA1338mT.jpg
 address | marker       |  length | data
       0 | 0xffd8 SOI
       2 | 0xffe1 APP1  |   46459 | Exif..II*.....................
```

```
   46463 | 0xffe3 APP3  |   65535 | ..............................   <--- what is this?
  112000 | 0xffe4 APP4  |   65535 | ..Hc..w .8<...z..M.77.h...{.....   <--- or this?
  177537 | 0xffe5 APP5  |    7243 | .U......K..u=).pl.W.F...B.$.3...
  184782 | 0xffdb DQT   |     132
  184916 | 0xffc0 SOF0  |      17
  184935 | 0xffc4 DHT   |      75
  185012 | 0xffda SOS
703 rmills@rmillsmbp:~/gnu/exiv2/trunk $
```

I'm willing to implement "Exif data exceeds 64k" feature in the future ***provided*** we have a more substantial specification and good pedigree test files.

#2 - 13 Dec 2018 13:00 - Robin Mills

I've looked at the following page: https://www.sno.phy.queensu.ca/~phil/exiftool/standards.html

Phil says: *Allow EXIF to span multiple JPEG segments. Multiple consecutive EXIF segments are simply concatenated into a single data block when reading. [Apparently Adobe has been doing this for years[11], and this ability was added to ExifTool in version 10.97.]*

[11] = XMP Specification part 3 November 2014, p. 13

#3 - 18 Dec 2018 14:50 - Phil Harvey

Hi Robin,

ExifTool will combine the data from adjacent EXIF segments unless subsequent segments start with a valid TIFF header ("MM\0\x2a" or "II\x2a\0") after the EXIF header ("Exif\0\0"). In case there is a TIFF header, I assume that we are dealing with multiple separate EXIF segments (which shouldn't happen, but I have seen it in the past). Due to this, ExifTool will write an EXIF segment which is one byte smaller if it would otherwise divide a >64kB EXIF block at a boundary where there just happens to be a TIFF header byte sequence. So, at least for ExifTool-written multi-segment EXIF, you can't just check that the first EXIF segment is the maximum size when deciding if it is multi-segment EXIF.

#4 - 18 Dec 2018 17:57 - Robin Mills

- File multi-segment_exif.jpg added

- Target version changed from 1.0 to 0.28

This looks like straightforward to implement and we'll add this early in v0.28. I've copied your test file

```
794 rmills@rmillsmbp:~/temp $ exiv2 -pS http://exiv2.dyndns.org:8080/userContent/testfiles/1232/multi-segment_
exif.jpg
STRUCTURE OF JPEG FILE: http://exiv2.dyndns.org:8080/userContent/testfiles/1232/multi-segment_exif.jpg
 address | marker       |  length | data
       0 | 0xffd8 SOI
       2 | 0xffe1 APP1  |   65535 | Exif..II*.....................
   65539 | 0xffe1 APP1  |    5603 | Exif..........................
   71144 | 0xffdb DQT   |     132
   71278 | 0xffc4 DHT   |     418
   71698 | 0xffc0 SOF0  |      17
   71717 | 0xffda SOS
795 rmills@rmillsmbp:~/temp $
```

I think I've understood what you've said about 2^16-1 to ensure that a tag doesn't straddle the jpeg chunks.

#5 - 19 Dec 2018 12:48 - Phil Harvey

- File cs4_extended_exif.jpg added

Hi Robin,

You requested a "reliable example" of multi-segment Exif. The "multi-segment_exif.jpg" I gave you was written by ExifTool. I have also attached a "cs4_extended_exif.jpg" sample written by Photoshop CS4. To generate this sample I filled the Exif ImageDescription with a very long bit of text (the exiftool source code). Unfortunately, as you can see, the Adobe software is buggy and writes an incorrect count for this tag, but at least it shows you how they intended to split the Exif into multiple segments.

The most difficult thing about adding support for this in ExifTool was in patching the -htmlDump feature to allow for gaps in the middle of the Exif due to the JPEG segment headers. Other than this, it was a fairly trivial addition.

#6 - 19 Dec 2018 12:49 - Phil Harvey

Phil Harvey wrote:

> Hi Robin,
>
> You requested a "reliable example" of multi-segment Exif. The "multi-segment_exif.jpg" I gave you was written by ExifTool. I have also attached a "cs4_extended_exif.jpg" sample written by Photoshop CS4. To generate this sample I filled the Exif ImageDescription with a very long bit of

text (the exiftool source code). Unfortunately, as you can see, the Adobe software is buggy and writes an incorrect count for this tag, but at least it shows you how they intended to split the Exif into multiple segments.

The most difficult thing about adding support for this in ExifTool was in patching the -htmlDump feature to allow for gaps in the middle of the Exif due to the JPEG segment headers. Other than this, it was a fairly trivial addition.

#7 - 19 Dec 2018 12:50 - Phil Harvey

(so much for my attempt to fix a typo in my original post)

#8 - 19 Dec 2018 13:59 - Robin Mills

Thanks, Phil. Looks interesting. I see you're also using extended XMP which is another item on our "one day" list.

```
829 rmills@rmillsmbp:~/Google Drive $ exiv2 -pS http://dev.exiv2.org/attachments/download/1259/cs4_extended_ex
if.jpg
STRUCTURE OF JPEG FILE: http://dev.exiv2.org/attachments/download/1259/cs4_extended_exif.jpg
 address | marker       |  length | data
       0 | 0xffd8 SOI
       2 | 0xffe0 APP0  |      16 | JFIF............
      20 | 0xffe1 APP1  |   65498 | Exif..MM.*..................n....
   65520 | 0xffe1 APP1  |   65498 | Exif..g keys we require'd
  131020 | 0xffe1 APP1  |   52820 | Exif..)            if ($$segData
  183842 | 0xffed APP13 |    4440 | Photoshop 3.0.8BIM.............
  188284 | 0xffe1 APP1  |    4323 | http://ns.adobe.com/xap/1.0/.<?x
  192609 | 0xffe1 APP1  |   65477 | http://ns.adobe.com/xmp/extensio
  258088 | 0xffe1 APP1  |   65477 | http://ns.adobe.com/xmp/extensio
  323567 | 0xffe1 APP1  |   56466 | http://ns.adobe.com/xmp/extensio
  380035 | 0xffe2 APP2  |    3160 | ICC_PROFILE......HLino....mntrRG chunk 1/1
  383197 | 0xffee APP14 |      14 | Adobe.d.......
  383213 | 0xffdb DQT   |     132
  383347 | 0xffc0 SOF0  |      17
  383366 | 0xffdd DRI   |       4
  383372 | 0xffc4 DHT   |     319
  383693 | 0xffda SOS
830 rmills@rmillsmbp:~/Google Drive $
```

#9 - 19 Dec 2018 14:56 - Phil Harvey

Yes. I generated this test file way back in 2008 by using Photoshop CS4 to write long values for IPTC:Caption-Abstract, EXIF:ImageDescription and XMP-dc:Description. This file was from when I added the ability for ExifTool to handle multi-segment XMP. Adobe was also writing multi-segment EXIF way back then, but I didn't notice it at the time because it didn't write the correct count for ImageDescription.

#10 - 19 Dec 2018 15:00 - Phil Harvey

I should mention that I don't remember ever seeing multi-segment EXIF in the wild. However, I have seen multi-segment XMP on a number of occasions.

Files

XmpPart3-2016-page13.png	142 KB	26 Sep 2016	Robin Mills
multi-segment_exif.jpg	4.9 MB	18 Dec 2018	Robin Mills
cs4_extended_exif.jpg	378 KB	19 Dec 2018	Phil Harvey