Exiv2 - Bug #961

Crash in digikam while reading metadata from a .MTS movie file

08 Jun 2014 12:02 - Robin Mills

Status:	Closed	**Start date:**	08 Jun 2014
Priority:	Normal	**Due date:**	
Assignee:	Robin Mills	**% Done:**	100%
Category:	video	**Estimated time:**	0.00 hour
Target version:	0.25		

Description

This is topic 1737 from the forum. http://dev.exiv2.org/boards/3/topics/1737

Related issues:

Related to Exiv2 - Feature #956: Support for .MTS movie format	**Closed**	**22 May 2014**	
Related to Exiv2 - Bug #963: Digikam is crashing for the image formats that i...	**Closed**	**08 Jun 2014**	**11 Jun 2014**

History

#1 - 10 Jun 2014 22:57 - Robin Mills

I don't see a bug here. Exiv2 is throwing an exception.

```
1014 rmills@rmills-ubuntu:~/gnu/exiv2/trunk $ exiv2 -v ../bugs/00056.MTS
File 1/1: ../bugs/00056.MTS
Exiv2 exception in print action for file ../bugs/00056.MTS:
../bugs/00056.MTS: The file contains data of an unknown image type
1015 rmills@rmills-ubuntu:~/gnu/exiv2/trunk $
```

#2 - 10 Jun 2014 23:17 - Robin Mills

I've had a look at the trace in http://dev.exiv2.org/attachments/610/digikam_crash_dump_05.txt and it looks like an unhandled exception:

```
Catchpoint 1 (exception thrown), __cxxabiv1::__cxa_throw (obj=0x7fff58004600,
130
    tinfo=0x3a3aa980a0 <typeinfo for Exiv2::BasicError<char>>,
131
    dest=0x33e6e9f030 <Exiv2::BasicError<char>::~BasicError()>)
132
    at ../../../../libstdc++-v3/libsupc++/eh_throw.cc:63
133
63      PROBE2 (throw, obj, tinfo);
134
(gdb) #0  __cxxabiv1::__cxa_throw (obj=0x7fff58004600,
135
    tinfo=0x3a3aa980a0 <typeinfo for Exiv2::BasicError<char>>,
136
    dest=0x33e6e9f030 <Exiv2::BasicError<char>::~BasicError()>)
137
    at ../../../../libstdc++-v3/libsupc++/eh_throw.cc:63
138
#1  0x00000033e6ee54a6 in Exiv2::ImageFactory::open(std::string const&) ()
139
   from /usr/lib64/libexiv2.so.13
```

Here's the code in Exiv2:

```
    Image::AutoPtr ImageFactory::open(const std::string& path)
    {
        BasicIo::AutoPtr io(new FileIo(path));
        Image::AutoPtr image = open(io); // may throw
        if (image.get() == 0) throw Error(11, path);
        return image;
    }
```

and the error message:

```
error.cpp:        { 11, N_("%1: The file contains data of an unknown image type") }, // %1=path
```

#3 - 11 Jun 2014 09:24 - Robin Mills

I've consolidated discussion of this from Issue #956 and Forum Topic #1737 into here.

Gilles has provided the following very helpful information in 1737:

```
From the backtrace, we can see :

#1 0x00000033e6ee54a6 in Exiv2::ImageFactory::open(std::string const&) () from /usr/lib64/libexiv2.so.13
#2 0x0000003a3a82956a in KExiv2Iface::KExiv2::load (this=0x7fff58389f20, filePath=...)
at /home/shipman/Downloads/dk/extra/libkexiv2/libkexiv2/kexiv2.cpp:298
#3 0x0000003a3bb15cf9 in Digikam::DMetadata::load (this=0x7fff58389f20, filePath=...)
at /home/shipman/Downloads/dk/core/libs/dmetadata/dmetadata.cpp:110

#1 is call from digiKam core;
#2 is call relevant in libkexiv2
#3 is call into Exiv2.

Look code from libkexiv2, and you will see that Exiv2 call is wrapped in C++ exception handler.
```

https://projects.kde.org/projects/kde/kdegraphics/libs/libkexiv2/repository/revisions/master/entry/libkexiv2/kexiv2.cpp#L298

And he has defended the digikam exception handling in libkexif in #956 by saying:

> digiKam itself do not have C++ exception, but it never use Exiv2 directly. For code design, all pass through libkexiv2, which is a C++ wrapper. All methods from libkexiv2 use C++ exception...
>
> So typically, digiKam must never crash due to a C++ exception from Exiv2...
>
> —

I'm not certain the handler in libkexiv2 is bullet proof. It only checks for Exiv2 errors. So, if the Exiv2 Error code was to throw an exception (such as stack-overflow or out-of-memory) it wouldn't be caught.

It's not easy to deal with this without recreating the environment with libkexiv2 wrapper and DigiKam present. I'm not trying to "pass the buck", however maybe Gilles could augment the exception handler in libkexiv2 to determine if an exception is being thrown by something else. I'm wondering if Exiv2 dies while trying to throw error 11.

Are there exceptions that cannot be caught? For example stack overflow. Maybe we've disappeared into a black hole of some kind.

There's another issue (#960) which involves memory corruption while opening files. This evening I will investigate that and perhaps #960 and #961 are the same issue.

#4 - 11 Jun 2014 10:01 - Gilles Caulier

Yes sure, libkexiv2 wrapper is dedicated to play properly with Exiv2 exception, without to touch digiKam code.

As all Exception wrapper is on private implementation, touching this will not break Binary compatibility of library.

Currently we have this king of implementation everywhere :

```
try {
// Exiv2 C++ exception calls here.
}
catch( Exiv2::Error& e ) {
d->printExiv2ExceptionError("Cannot load metadata from file ", e);
}
```

What do you want that i change here in this exception handling exactly ?

Gilles Caulier

#5 - 11 Jun 2014 10:54 - Robin Mills

Thanks for getting back to me, Gilles. How about:

```
try {
  // code here
}
catch( Exiv2::Error& e ) {
    d->printExiv2ExceptionError("Cannot load metadata from file ", e);
```

```
}
catch (...) {
   cout << "default exception";
}
```

It is possible that printExiv2ExceptionError() is throwing an exception! So the simplest code might be:

```
try {
  // code here
}
catch (...) {
   cout << "default exception";
}
```

Robin

#6 - 11 Jun 2014 13:53 - Gilles Caulier

Done with this commit in libkexiv2 :

http://commits.kde.org/libkexiv2/8a8dc535d504b70776677b933ac761bba0f7a4ae

I closed also this KDE bugzilla entry :

https://bugs.kde.org/show_bug.cgi?id=331679

Gilles Caulier

#7 - 11 Jun 2014 14:17 - Robin Mills

- Status changed from Assigned to Resolved

Thanks, Gilles.

It's always nice to work with you. I'll mark #961 and #963 as "resolved" and they'll be closed during review prior to the release of 0.25.

Robin

#8 - 08 May 2015 16:27 - Robin Mills

- % Done changed from 0 to 100

#9 - 21 Jun 2015 16:41 - Andreas Huggel

- Status changed from Resolved to Closed