ctools - Bug #1823

Copy of ctbin tool fails on Linux if log file is opened

07/22/2016 12:47 AM - Knödlseder Jürgen

Status:	Closed	**Start date:**	07/22/2016
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.2.0		

Description

The following script leads to a segmentation fault on Linux (for example CentOS 6):

```
# Set-up ctbin
bin = ctools.ctbin()
bin['inobs']    = self._events
bin['outcube']  = 'cntmap.fits'
bin['ebinalg']  = 'LOG'
bin['emin']     = 0.1
bin['emax']     = 100.0
bin['enumbins'] = 20
bin['nxpix']    = 200
bin['nypix']    = 200
bin['binsz']    = 0.02
bin['coordsys'] = 'CEL'
bin['proj']     = 'CAR'
bin['xref']     = 83.63
bin['yref']     = 22.01
bin['logfile']  = 'ctbin_py1.log'
bin['chatter']  = 2

# Run ctbin tool
bin.logFileOpen()   # Make sure we get a log file
bin.run()

# Check content of observation and cube
self._check_observation(bin, 5542)
self._check_cube(bin.cube(), 5542)

# Test copy constructor
cpy_bin = bin.copy()    <=== Segmentation fault here
```

The segmentation fault disappears when the following line is commented out:

```
#       bin.logFileOpen()   # Make sure we get a log file
```

It therefore seems that there is a problem with copying a ctool that has an open log file.

As temporary fix, I will comment the critical line out.

History

#1 - 07/22/2016 12:52 AM - Knödlseder Jürgen

Sometimes, the job also just hangs.

#2 - 08/04/2016 08:50 PM - Knödlseder Jürgen

Here the output on CentOS 6:

./test_python_ctools.sh: line 25: 16514 Erreur de segmentation (core dumped) ./test_python_ctools.py

or it hangs or

```
*** glibc detected *** python: munmap_chunk(): invalid pointer: 0x00000000013873a8 ***
======= Backtrace: =========
/lib64/libc.so.6[0x318c0753c6]
/lib64/libc.so.6(_IO_free_backup_area+0x19)[0x318c073aa9]
/lib64/libc.so.6(_IO_file_close_it+0x33)[0x318c071c93]
/lib64/libc.so.6(fclose+0x178)[0x318c065b68]
/home/jenkins/jenkins/install/integrate/gammalib/lib/libgamma.so.2(_ZN4GLog5closeEv+0x25)[0x7ffc9e30d495]
/home/jenkins/jenkins/install/integrate/gammalib/lib/libgamma.so.2(_ZN12GApplication12free_membersEv+0x48)[0x7ffc9e3090f8]
/home/jenkins/jenkins/install/integrate/gammalib/lib/libgamma.so.2(_ZN12GApplicationD1Ev+0x24)[0x7ffc9e30cb84]
/home/jenkins/jenkins/workspace/ctools-integrate-os/label/centos6_64/pyext/build/ctools/ctools/_tools.so(+0x1262c)[0x7ffc9a8eb62c]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalFrameEx+0x4f82)[0x31980de5b2]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalFrameEx+0x62d5)[0x31980df905]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalCodeEx+0x904)[0x31980e05d4]
/usr/lib64/libpython2.6.so.1.0[0x319806e9e0]
/usr/lib64/libpython2.6.so.1.0(PyObject_Call+0x53)[0x3198043e13]
/usr/lib64/libpython2.6.so.1.0[0x31980592ef]
/usr/lib64/libpython2.6.so.1.0(PyObject_Call+0x53)[0x3198043e13]
/usr/lib64/libpython2.6.so.1.0(PyEval_CallObjectWithKeywords+0x43)[0x31980d8af3]
/home/jenkins/jenkins/install/integrate/gammalib/lib64/python2.6/site-packages/gammalib/_test.so(_ZN16GPythonTestSuite4testEv+0x5d)[0x7ffc9bdb
5a1d]
/home/jenkins/jenkins/install/integrate/gammalib/lib/libgamma.so.2(_ZN10GTestSuite3runEv+0x1ac)[0x7ffc9e32159c]
/home/jenkins/jenkins/install/integrate/gammalib/lib/libgamma.so.2(_ZN11GTestSuites3runEv+0x219)[0x7ffc9e3259b9]
/home/jenkins/jenkins/install/integrate/gammalib/lib64/python2.6/site-packages/gammalib/_test.so(+0xbab6)[0x7ffc9bda1ab6]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalFrameEx+0x4f82)[0x31980de5b2]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalFrameEx+0x62d5)[0x31980df905]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalCodeEx+0x904)[0x31980e05d4]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalFrameEx+0x526f)[0x31980de89f]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalCodeEx+0x904)[0x31980e05d4]
/usr/lib64/libpython2.6.so.1.0(PyEval_EvalCode+0x32)[0x31980e06d2]
/usr/lib64/libpython2.6.so.1.0[0x31980fb71c]
/usr/lib64/libpython2.6.so.1.0(PyRun_FileExFlags+0x90)[0x31980fb7f0]
/usr/lib64/libpython2.6.so.1.0(PyRun_SimpleFileExFlags+0xdc)[0x31980fcc8c]
/usr/lib64/libpython2.6.so.1.0(Py_Main+0xadd)[0x31981092ed]
/lib64/libc.so.6(__libc_start_main+0xfd)[0x318c01ecdd]
```

The line that produces the problem is

```
cpy_bin = bin.copy()
```

#3 - 08/04/2016 10:01 PM - Knödlseder Jürgen

It seems that the problem is related to the GLog::copy_members() method that simply copies the log file descriptor:

```
void GLog::copy_members(const GLog& log)
{
   // Copy attributes
   m_max_length = log.m_max_length;
   m_indent     = log.m_indent;
   m_stdout     = log.m_stdout;
   m_stderr     = log.m_stderr;
   m_use_date   = log.m_use_date;
   m_linestart  = log.m_linestart;
   m_file       = log.m_file;
   m_filename   = log.m_filename;
   m_name       = log.m_name;
   m_buffer     = log.m_buffer;
   m_chatter    = log.m_chatter;

   // Return
   return;
}
```

Now there are two instances of the same file pointer existing, and one class may close the file while the other still has a pointer to the already closed file and wants to close it again. This can be solved by duplicating the file pointer in GLog::copy_members() as follows:

```
void GLog::copy_members(const GLog& log)
{
   // Copy attributes
   m_max_length = log.m_max_length;
   m_indent     = log.m_indent;
   m_stdout     = log.m_stdout;
   m_stderr     = log.m_stderr;
   m_use_date   = log.m_use_date;
   m_linestart  = log.m_linestart;
   m_filename   = log.m_filename;
   m_name       = log.m_name;
   m_buffer     = log.m_buffer;
   m_chatter    = log.m_chatter;

   // Copy file pointer by duplicating the file descriptor and opening
   // the duplicated file descriptor
   if (log.m_file != NULL) {
      m_file = fdopen(dup(fileno(log.m_file)), "a");
   }
   else {
      m_file = NULL;
   }

   // Return
   return;
}
```

#4 - 08/04/2016 10:56 PM - Knödlseder Jürgen

- Status changed from New to Closed

- Assigned To set to Knödlseder Jürgen

- Target version set to 1.2.0

Fixed and merged into devel.

#5 - 03/03/2017 10:51 AM - Knödlseder Jürgen

- % Done changed from 0 to 100