GammaLib - Change request #1666

Use GFilename instead of std::string for file name arguments.

02/13/2016 10:50 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	02/13/2016
Priority:	Urgent	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.1.0		

Description

The GFilename class should replace everywhere the std::string argument for file name arguments.

History

#1 - 02/13/2016 10:52 PM - Knödlseder Jürgen

- % Done changed from 0 to 10

Implemented a Python typemap that allows transparent usage of GFilename, enabling providing still a string as argument:

```
%typemap(in) GFilename& (GFilename temp) {
   if (PyString_Check($input)) {
      temp = GFilename(std::string(PyString_AsString($input)));
      $1 = &temp;
   }
   else {
      void *filename_argp1 = 0;
      if (SWIG_IsOK(SWIG_ConvertPtr($input, &filename_argp1, SWIGTYPE_p_GFilename, 0))) {
         $1 = reinterpret_cast<GFilename*>(filename_argp1);
      }
      else {
         SWIG_exception(SWIG_TypeError, "GFilename expected");
      }
   }
}
```

#2 - 02/13/2016 11:03 PM - Knödlseder Jürgen

- % Done changed from 10 to 20

Use GFilename in GEbounds and GGti (to check whether the interface works seamless).

#3 - 02/14/2016 03:33 PM - Knödlseder Jürgen

There is still a problem with the Python interface, specifically with filename constructors. The SWIG code generated for overloaded constructors is the following:

```
SWIGINTERN PyObject *_wrap_new_GFits(PyObject *self, PyObject *args) {
  int argc;
  PyObject *argv[3];
  int ii;

  if (!PyTuple_Check(args)) SWIG_fail;
  argc = args ? (int)PyObject_Length(args) : 0;
  for (ii = 0; (ii < 2) && (ii < argc); ii++) {
    argv[ii] = PyTuple_GET_ITEM(args,ii);
  }
  if (argc == 0) {
    return _wrap_new_GFits__SWIG_0(self, args);
  }
  if (argc == 1) {
    int _v;
    int res = SWIG_ConvertPtr(argv[0], 0, SWIGTYPE_p_GFilename, 0);
    _v = SWIG_CheckState(res);
    if (_v) {
      return _wrap_new_GFits__SWIG_2(self, args);
    }
  }
  if (argc == 1) {
    int _v;
    int res = SWIG_ConvertPtr(argv[0], 0, SWIGTYPE_p_GFits, 0);
    _v = SWIG_CheckState(res);
    if (_v) {
      return _wrap_new_GFits__SWIG_3(self, args);
    }
  }
  if (argc == 2) {
    int _v;
    int res = SWIG_ConvertPtr(argv[0], 0, SWIGTYPE_p_GFilename, 0);
    _v = SWIG_CheckState(res);
    if (_v) {
      {
        int res = SWIG_AsVal_bool(argv[1], NULL);
        _v = SWIG_CheckState(res);
      }
      if (_v) {
        return _wrap_new_GFits__SWIG_1(self, args);
      }
    }
  }
}
```

At the level of this code there is no check for a constructor where GFilename has been replaced by a string, and consequently the constructor throws an exception. I need to find a method to add constructor code for a string.

#4 - 02/14/2016 04:36 PM - Knödlseder Jürgen

I needed to add a %typecheck directive (typecheck typemap). Note that I also found out that the $1_descriptor should be used instead of a hard-wired pointer. The code now looks like this:

```
%typemap(in) GFilename& (GFilename temp) {
   if (PyString_Check($input)) {
       temp = GFilename(std::string(PyString_AsString($input)));
       $1 = &temp;
   }
   else {
      void *filename_argp1 = 0;
      if (SWIG_IsOK(SWIG_ConvertPtr($input, &filename_argp1, $1_descriptor, 0))) {
         $1 = reinterpret_cast<GFilename*>(filename_argp1);
      }
      else {
         SWIG_exception(SWIG_TypeError, "GFilename expected");
      }
   }
}
%typecheck(SWIG_TYPECHECK_DOUBLE) GFilename& {
   if (PyString_Check($input) ||
      SWIG_CheckState(SWIG_ConvertPtr($input, 0, $1_descriptor, 0))) {
      $1 = 1;
   }
   else {
      $1 = 0;
   }
}
```

#5 - 02/16/2016 03:13 AM - Knödlseder Jürgen

- Status changed from In Progress to Closed

- % Done changed from 20 to 100

All filename arguments have been replaced by GFilename.

Please note that the gammalib::file_exists() and gammalib::is_fits() methods have been removed as the methods are now replaced by GFilename::exists() and GFilename::is_fits() (there is also a GFilename::remove() method).

Merged into devel.

#6 - 02/17/2016 05:02 PM - Mayer Michael

I don't know if that happened here, but the FITS selection syntax does not work anymore:

```
import gammalib
f1 = gammalib.GFits("$CTOOLS/test/data/crab_events.fits.gz")
print(f1["EVENTS"].nrows())
f2 = gammalib.GFits("$CTOOLS/test/data/crab_events.fits.gz[EVENTS][ENERGY>5.0]")
print(f2["EVENTS"].nrows())
```

This code will return

```
6141
6141
```

Any idea what went wrong?

#7 - 02/17/2016 05:38 PM - Knödlseder Jürgen

- Status changed from Closed to In Progress

- Priority changed from Normal to Urgent

- % Done changed from 100 to 80

user#77 wrote:

> I don't know if that happened here, but the FITS selection syntax does not work anymore:
> [...]
> This code will return
> [...]
> Any idea what went wrong?

I probably made a mistake when reworking the code. I will look into that.

#8 - 02/18/2016 01:02 AM - Knödlseder Jürgen

- Status changed from In Progress to Feedback

I think this is fixed now. Can you checkout devel and see if it works as expected?

#9 - 02/18/2016 08:45 AM - Mayer Michael

Yes works just fine again - thank you.

#10 - 02/18/2016 09:24 AM - Knödlseder Jürgen

- Status changed from Feedback to Closed

- % Done changed from 80 to 100

Close now again.