GammaLib - Feature #1733

Handle >3 dimensional maps in GSkyMap

03/05/2016 05:32 PM - Knödlseder Jürgen

Status:	Closed		**Start date:**	03/05/2016
Priority:	Normal		**Due date:**	
Assigned To:	Knödlseder Jürgen		**% Done:**	100%
Category:			**Estimated time:**	0.00 hour
Target version:	1.1.0			

Description

The GTuple class should implement an integer tuple that can be used to pass an arbitrary number of indices as argument to a method. Use cases for this class are the map argument of the GSkyMap class to allow the creation of n-dimensional map cubes, and the FITS image pixel access operators.

Here a draft of the class:


```
class GTuple : public GBase {
public:
   // Constructors
   GTuple(void);
   GTuple(const int& i1);
   GTuple(const int& i1, const int& i2);
   GTuple(const int& i1, const int& i2, const int& i3);
   GTuple(const std::vector<int>& index);

   // Operators
   int&      operator[](const int& i);
   const int& operator[](const int& i) const;

   // Methods
   int size(void) const

protected:
   // Data members
   int       m_size;
   int       m_index1;
   int       m_index2;
   int       m_index3;
   std::vector<int> m_index;
```


History

#1 - 05/29/2016 10:37 PM - Knödlseder Jürgen

I'm not fully sure that a GTuple class is indeed needed. An alternative approach would be to use a variadic function (see http://en.cppreference.com/w/cpp/utility/variadic) to define an arbitrary number of additional dimensions in GSkyMap. The advantage of this approach would be that the current GSkyMap interface could be basically kept, and that GSkyMap would have additional methods for higher map dimensions.

Here some C++ use cases that use an extension of the current format:


```
GSkyPixel pix;
GSkyDir   dir;
GSkyMap   map("CAR", "CEL", 0.0, 0.0, 0.1, 0.1, 100, 100, 10, 10);
double    value1 = map(pix,0,1);
double    value2 = map(dir,0,1);
```


and Python use cases

```
pix    = gammalib.GSkyPixel()
dir    = gammalib.GSkyDir()
map    = gammalib.GSkyMap("CAR", "CEL", 0.0, 0.0, 0.1, 0.1, 100, 100, 10, 10)
value1 = map(pix,0,1)
value2 = map(dir,0,1)
```

#2 - 05/29/2016 10:38 PM - Knödlseder Jürgen

- Subject changed from Create GTuple class to Handle >3 dimensional than maps in GSkyMap

#3 - 05/29/2016 11:03 PM - Knödlseder Jürgen

- Subject changed from Handle >3 dimensional than maps in GSkyMap to Handle >3 dimensional maps in GSkyMap

#4 - 05/30/2016 11:43 AM - Knödlseder Jürgen

It's maybe sufficient (at least for now) to add a shape() method, along the logic that numpy uses, to define the shape of the maps. For example

GSkyMap map("CAR", "CEL", 0.0, 0.0, 0.1, 0.1, 100, 100, 10);
map.shape(5,2);

would organise the 10 maps in the objects into a two-dimensional array with axes lengths of 5 times 2 (the shape() method would throw an exception in case that the total number of maps in the GSkyMap object is not equal to the product of the shape arguments.

When GSkyMap loads a map it will automatically determine the shape from the FITS file dimension, and compute the total number of maps as the product of the shape parameters.

A method ndim() should be added that returns the dimension of the shape part (in analogy to numpy again).

#5 - 05/30/2016 10:09 PM - Knödlseder Jürgen

- Status changed from New to Closed

- Assigned To set to Knödlseder Jürgen

- Target version set to 1.1.0

- % Done changed from 0 to 100

The change has been implemented, unit tests have been added, is now in devel.