GammaLib - Feature #1564

Implement setter method for GSkyRegion::name()

11/02/2015 05:09 PM - Mayer Michael

Status:	Closed	**Start date:**	11/02/2015
Priority:	Normal	**Due date:**	
Assigned To:	Mayer Michael	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Currently, it is only possible to retrieve the name of a GSkyRegion object. A setter for this property is needed to create several GSkyRegion objects (e.g. in python) and append them to a GSkyRegions container.

The following code won't work without the setter method and will throw an exception:

```
import gammalib
regions = gammalib.GSkyRegions()
for i in range(5):
    region = gammalib.GSkyRegionCircle(83.6,22.0,0.2*i)
    # region.name("Region_"+str(i)) <-- This needs to be implemented
    regions.append(region)
```

History

#1 - 11/02/2015 09:17 PM - Knödlseder Jürgen

I agree that such a method should be added.

I do not recall why we have decided that sky regions need to have a name. Maybe we should revisit this issue; at least check why at the time we have implemented the sky regions we decided that each region should have a name.

#2 - 11/03/2015 08:16 AM - Mayer Michael

I went through the respective issues. Especially in #535, there was a long discussion about the interface of the GSkyRegion class. In comment 23 you asked about the purpose of the name member. Pierrick replied he replaced this attribute by a m_type member. Apparently this did not happen.
I guess after all, we kept this attribute to allow the following access from the container by name: GSkyRegions::operator[](std::string name]. This makes sense to me. because the user does not necessarily require the id to access the container element. Drawback: every region the requires a name, different from "" which is the default.

#3 - 11/03/2015 12:36 PM - Knödlseder Jürgen

I thought about this a bit more.

We could allow the name to be non-unique. We would then add an integer argument to the operator (that defaults to 0) to access the elements by name. In that way the ith element with a given name can be accessed. At the same time, it does not hurt to leave all names blank (i.e. if one does not want to use the names).

#4 - 11/03/2015 01:50 PM - Mayer Michael

If I understand correctly you mean to change the existing operator in GSkyRegions.hpp

GSkyRegion* operator[](const std::string& name);

into:

GSkyRegion* operator[](const std::string& name, const int& index = 0);

which would imply non-unique region names.
To implement the changes, we further would need to adapt the following methods:

- GSkyRegion* set(const std::string& name, const GSkyRegion& region);
- void remove(const std::string& name);
- GSkyRegion* insert(const std::string& name, const GSkyRegion& region);

The above functions should also take an additional index as argument, with default value 0, right?

#5 - 11/05/2015 06:11 PM - Knödlseder Jürgen

I had not recognized before that name is so widely used.

I'm still wondering about the use cases of the access by name (the reason I'm hesitating is because this now will make the interface overly complex).

#6 - 11/05/2015 08:33 PM - Mayer Michael

About the use case: I agree the interface would become too complex. I don't really see the need that every region needs a unique name.
We could instead just keep everything as it is and only make the name attribute optional. The operator[](std::string) could then return the first occurrence of the given name. E.g. if every region name is empty, regions[""] would return the first element.

#7 - 11/05/2015 09:47 PM - Knödlseder Jürgen

Mayer Michael wrote:

> About the use case: I agree the interface would become too complex. I don't really see the need that every region needs a unique name.
> We could instead just keep everything as it is and only make the name attribute optional. The operator[](std::string) could then return the first occurrence of the given name. E.g. if every region name is empty, regions[""] would return the first element.

Agree. We may even think about dropping access by name (and also set, insert and remove). I think this was carried over from other classes, but I'm still not convinced that the name of a region has such a high relevance to merit own access functions (one can always loop over all regions to find a region with a given name). I somehow get the feeling that we make a simple thing complicated.

#8 - 11/06/2015 09:56 AM - Mayer Michael

- Status changed from New to Pull request

- Assigned To set to Mayer Michael

- % Done changed from 0 to 100

I realised that in the python extension GSkyRegions.i the operator[](std::string) is not supported anyway. I hence followed your suggestion and removed the following functions:

- set(std::string, GSkyRegion)
- insert(std::string, GSkyRegion)
- remove(std::string)
- operator[](std::string)
- contains(std::string)
- get_index(std::string)

I recompiled and run make check to not accidentally break anything. It seems the name for sky regions was not used in gammalib at all, so everything is works fine.

The changes are available on branch *1564-allow-nonunique-SkyRegions*

#9 - 11/06/2015 10:50 AM - Knödlseder Jürgen

- Status changed from Pull request to Closed

Merged into devel.