Subject: Re: Is there a way to use C++ code in IDL
Posted by Paul Van Delst[1] on Thu, 20 Feb 2003 19:28:06 GMT
View Forum Message <> Reply to Message

CelticBlues wrote:
>
> The file is a home grown tagged file format containing a type of image data
> and additional information about the conditions at the time the image was
> taken etc. Here is an example of what a simple file may look like:
> ----------
> [System]
> Name = 1
> [End]
> [Data Sets] // will be at least one data set, could be as many as 3
> DataSet1
> [End]
> [Files Attributes]
> [Conditions File]
> [Properties]
> Name = 97918.10
> CreateTime = Thu Sept 18 1997 at 11:21:36
> LastAccessTime = Mon Jan 01 1601
> LastModifiedTime = Thu Sept 18 1997 at 11:21:36
> Size = 16882000 // bytes
> [End]
> [Frame information]
> Width = 133
> Height = 148
> [End]
> [End] // end [Telemetry File]
>
> [Saved File]
> CreateDate = Tue Feb 11 2003 at 16:08:54
> Width = 133
> Height = 148
> FrameNumber = 1
> DataUnits = meters
> Comment =
> CoordinateSystem = UTM // Latlong or UTM
> [End]
>
> [End] // end [Files Attributes]
>
> [DataBlock:DataSet1]
> R1P1 R1P2...R1P148
> R2P1 R2P2...R2P148
> .
> .

>   .
>   R133P1 R133P2...R133P148
> [End]
>
> ----------
>
> R1P1 = Row 1 Pixel 1, R2P1 = Row 2, Pixel 2 etc.
>
> One additional bit of info... the datablock may not actually have CR/NL
> after each row, i.e. it may be
>
> [DataBlock:DataSet1]
>   R1P1 R1P2...R1P148 R2P1 R2P2...R2P148 ... R133P1 R133P2...R133P148
> [End]
>
> Ok, assume that I want to rewrite the parser in IDL... From what I
> understand it is simple enough to read an ascii table from a file, but what
> about reading this type of format?

The reading isn't the problem - it's the sorting of the read-in bits. :o)

But, it doesn't look too difficult. Given what you know about the allowable syntax in your ASCII file, you can create the structures up front, e.g.

```
  Properties = { Name:             0.0, $
                 CreateTime:       ' ', $
                 LastAccessTime:   ' ', $
                 LastModifiedTime: ' ', $
                 Size:             0L }

  FrameInformation = { Width:  0L, $
                       Height: 0L }

  ConditionsFile = { Properties: Properties, $
                     Frame_Information: FrameInformation }

  SavedFile = { ...etc }


  FilesAttributes = { ConditionsFile: ConditionsFile, $
                      SavedFile: SavedFile }
```

etc..etc.

although your lines may get a bit long, e.g.

```
  print, FilesAttributes.ConditionsFiles.Properties.LastModifiedTime
```

...phew!

I would use structure contructor functions (search for "Automatic Structure Definition" in the IDL help) to do it, so you can automatically define a valid structure containing all the allowed bits. You could also approach this in an OO way (search for "IDL Object Overview"). I'm not an IDL OO feller, so others will have more to say about that aspect.

All the data stuff would be stored in a pointer array structure element that you can allocate when you determine what the required dimensions are.

Actually it sounds like a neato little project to do this. you'll learn everything you ever wanted to know about IDL structures, and if you're keen, about the OO capabilities also.

paulv

--
Paul van Delst
CIMSS @ NOAA/NCEP/EMC
Ph: (301)763-8000 x7274
Fax:(301)763-8545
