Subject: Looking for IDL code documentation standards
Posted by wcramer on Mon, 16 May 2005 14:03:46 GMT
View Forum Message <> Reply to Message

Is there somewhere in the IDL guides or on the web that the IDL code documentation standards are located? I've been able to piece together some of it from IDL code examples, but I'd like to see a definitive guide.

Thanks,
Doug

Subject: Re: Looking for IDL code documentation standards
Posted by savoie on Tue, 17 May 2005 15:53:31 GMT
View Forum Message <> Reply to Message

Ben Panter <me@privacy.net> writes:

> Antonio Santiago wrote:
>> Ken Mankoff wrote:
>
>>>
>>> I use this to generate pretty-print PS files. I think a2ps can also be
>>> used to generate HTML.
>>>
>>> a2ps --pre=idl --pro=color foo.pro
>>>
>>> There is also a pretty-print-buffer command in emacs.
>>>
>>> -k.
>>>
>> I use htmlize-buffer (http://fly.srk.fer.hr/~hniksic/emacs/htmlize.el) with
>> color-themes.

I use ps-print-buffer-with-faces.

Matt

--
Matthew Savoie - Scientific Programmer
National Snow and Ice Data Center
(303) 735-0785 http://nsidc.org

Subject: Re: Looking for IDL code documentation standards

>> Reserved Words - lowercase (e.g. begin, end, do, for, if, while)
>>
>> Variables: lowercase, words separated by "_" (e.g. my_var)
>> Routines: lowercase, words separated by "_" (e.g. my_func)
>> Keywords: UPPERCASE, words separated by "_" (e.g. MY_KEYWORD)
>>
>> Classes: MixedCase, no separation character (e.g. MyWonderfulClass)
>> Methods: MixedCase, no separation character (e.g. GetProperty)
>
>
> I find it useful to distinguish, in my code, names of routines that are
> part of the IDL distribution from names of contributed routines. I do this
> in the following way:
>
> PRINT, SIN(MyFunc(x))
>
> and apologize to all those who detest the all-uppercase way. What I
> advocate is just only to make clear in *some* way that "sin" is part of
> the IDL distribution, whereas "myfunc" isn't.
>
> This helps avoid confusion and makes it easier to track down problems
> caused by collisions in that small world called namespace.

Makes perfect sense. One of the rules that I didn't elaborate on was that I prepend a short "namespace" to the front of my functions and procedures. If I have a project called "Mike's Cool Project," the typical routine name would look like something like mcp_function or mcp_procedure. Any objects related to the project have the same convention (e.g. MCPgrCoolTextThingy).

-Mike