Subject: Re: Compiling file with many functions: huge performance difference between IDL and IDLDE
Posted by mvukovic on Wed, 17 Mar 2004 20:16:02 GMT
View Forum Message <> Reply to Message

Sidney Cadot <sidney@jigsaw.nl> wrote in message news:<1079516867.600179@euler.servers.luna.net>...
> Hi all,
>
> For a system we're making, a rather big IDL file is generated containing
> well over 12,000 function definitions, accompanied by a selector
> function (see below for a rationale).
>
> What we're seeing is that in command-line IDL, this works like a charm:
> compilation of the file takes about 4--5 seconds on a reasonably fast
> machine, which is acceptable.
>
> However, when this file is compiled from within IDLDE, this takes well
> over three minutes-- rougly a factor 60 increase(!)
>
> Does anybody know what causes this, and perhaps a solution?
>
> We tried pre-compiling the functions using a SAV file; this yields a
> significant increase both in IDL (cmd line version): 3 sec, and IDLDE
> (used time down to 87 seconds), but the relative difference is still
> quite puzzling.
>
> Best regards,
>
>   Sidney Cadot
>   Science and Technology Corp., The Netherlands
>
>
>
>
> P.S. the reason we're doing this is that we need to implement a
> string-based map with optiomal performance, like this:
>
```
> FUNCTION f_tom
>   RETURN, 123
> END
>
> FUNCTION f_dick
>   RETURN, 456
> END
>
> FUNCTION f_harry
>   RETURN, 789
>  END
>
>  FUNCTION f, name
>     CATCH, error_status
>     IF error_status EQ 0 THEN RETURN, -1
>     RETURN, call_function("f_" + name)
>  END
```

Out of curiosity, would a structure work here:
a={f_tom:123,f_dick:456,f_harry:789...} ?

It could be created using create_struct.

Retrieve info using

a=str.f_dick

Curious minds want to know :-)

(And never mind about ``curiosity kills the cat'')

Mirko