Subject: Re: Compiling file with many functions: huge performance difference between IDL and IDLDE
Posted by btt on Thu, 18 Mar 2004 20:50:56 GMT
View Forum Message <> Reply to Message

Sidney Cadot wrote:
> Mirko Vukovic wrote:
>
>>> 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 :-)
>
>
> Your idea is sound, but I am not aware of a way to retrieve the index of
> a tag-name based on its name.
>
> You assume that "f_dick" is available at compile time, whereas I need to
> resolve the string at runtime. Something like this would work:
>

> i = TAG_INDEX("f_dick", str)
> value = str.(i)
>
> ... But only if functionality to get a tag index can be retrieved from a
> struct (anyone knows how to do this?) and if its fast, i.e. if IDL
> implements it via a hash table or similar.
>

How about this ?

```
IDL> print, tag_index(['color', 'clip', 'banana'], !P, count = count) &
print, count
          4          3         -1
          2
```

```
;-----start
FUNCTION TAG_INDEX, tagName, str, count = count

tags = TAG_NAMES(str)

n = n_elements(tagName)

tag_index = LonArr(n, /noZero)

Count = 0L
For i = 0L, n-1 Do begin
 tag_index[i] = (WHERE(tags EQ StrUpCase(tagName[i]), cnt))[0]
 Count = Count + (cnt GT 0)
EndFor

Return, tag_index
END
;-----finish
```

Ben
