Subject: Re: compiling before needing
Posted by Todd Clements on Wed, 12 Sep 2001 21:54:21 GMT
View Forum Message <> Reply to Message

> My programm consists of several functions and procedures in
> corresponding files. Is there a way of automatically compiling all files
> (functions/procedures) before they are needed

Reimar is right - just stick a resolve_all at the beginning of the function that is called, and IDL will resolve all routines.

I've also written a routine that will do this much, much more manually (which isn't what you wanted, but I thought I'd share anyway). It will let you check if a routine is compiled and you can optionally force it to compile. The /compile keyword will compile it if it's not already compiled, and the /force_compile keyword will compile it even if it's already compiled. The function returns 1 if the routine is compiled and 0 if it is not.

Todd

```
;----------------------

function is_routine_compiled, routineName, compile=compile,$
                         force_compile=force_compile
compile_opt idl2

  if( typeof( routineName ) ne 7 ) then begin
     print, 'Must pass string to is_routine_compiled'
     return, 0
  endif

  rn = strupcase( routineName )

  a = routine_info()
  a = [a, routine_info(/functions)]
  junk = where( a eq rn )
  if( junk[0] ne -1 ) then begin
     ;; We may have found it, but it may not be resolved
     a = routine_info(/unresolved)
     a = [a, routine_info( /unresolved, /functions )]
     junk = where( a eq rn )
     if( junk[0] ne -1 ) then junk = -1 else junk = 1
  endif

  if( junk[0] eq -1 or keyword_set(force_compile) ) then begin
     if( keyword_set( compile ) or $
      keyword_set( force_compile )) then begin
```

```
        resolve_routine, rn, /compile_full_file, /either
        return, is_routine_compiled(rn)
      endif else return, 0
  endif else return, 1
end ;;is_routine_compiled
```