Subject: Re: something like perl's 'require 5.4'
Posted by Vapuser on Tue, 17 Apr 2001 20:44:19 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@cow.physics.wisc.edu> writes:

```
>  Vapuser <vapuser@catspaw.jpl.nasa.gov> writes:
>>    How about:
>>
>>     version=strupcase(strjoin(strsplit(!version.release,'[.-_]', /extract), ""))
>>     input_version=strupcase(strjoin(strsplit(desired_release,'[. -_]',/extract), ""))
>>    if version lt input_version then 'eeek'
>>
>>    Clearly `desired_release' would have to be input as a string.
>>
>>    Of couse, this method 'requires' IDL 5.3 (IIRC), to get the regular
>>    expression semantics of strsplit. The other question is will RSI use
>>    any separator in version designations besides these three?
>
>
>  Umm, I never thought I'd be saying this, but aren't these a bit
>  overengineered?  I have found that in 95% of the cases the DOUBLE
>  compare will work, and in 4% of the cases the STRING compare will
>  work.
>
```

When these two lines, which probably are too much for inline code,
are in their own procedure/function, I'd say 'no.'

I'll just have a 'require, "5.4"' as (one of) the first
things in any piece of code.

```
>  If you know which version number you are targetting, then clearly you
>  can fashion a !VERSION.RELEASE comparison that will do the job.
>
```

Probably so, but then why not just put it in a little procedure
which just fails to the command line?

Alternately, one could have a function like:

```
function collapseversion, version
  if n_elements(version) eq 0 then ERROR
   return,strupcase(strjoin(strsplit(version,'[.-_]',/extract), ""))
end
```

-- and do --

if collapseversion(!version.release) lt '54' then ERROR.

The best of both worlds.

> William, you were probably thinking of a writing general procedure to
> handle this type of version enforcement. It's actually a good idea.

Yep. My home software cache, which *was* built on my work stuff mostly, got nuked and so I've decided to just rebuild it from scratch. I'm in the utility writing section of the rebuild and finding myself wanting to use features that have come up since I wrote most of these utilites the first time, hence the need for this sort of checking.

And since a failure of this sort of requirement checking really is kinda fatal, i.e. this procedure/function calls stregex for which you *must* have idl.version >= 5.3, I don't see what the problem is. Put it in a proceduce and fail the sucker if the version isn't high enough.

Oh... wait. I just thought of something! Maybe you could branch around code that had some special dependencies.

No problem: write 2, a procedure that fails to the prompt and a function that returns true/false.

> But I've found that an "IF !VERSION.RELEASE..." is compact enough to
> go right into my code.

True. I guess I'm of a different bent. I write lots of utility functions/prodecures that I use all through my code, mainly because I get tired of typing their contents all the time.

But thanks for the suggestions.

whd
--
William Daffer: 818-354-0161: William.Daffer@jpl.nasa.gov