Subject: Testing validity of numerical content of a string
Posted by Y.T. on Thu, 07 Dec 2006 01:29:03 GMT
View Forum Message <> Reply to Message

Maybe this is a stupid question, but is there a way in IDL to tell whether a string contains a valid representation of a particular data type? For example an integer? I can do something like

test = fix(str)

which will return zero (and throw an error message) if str is, say, "hello". But is there a way to tell whether type conversion of a string will work if attempted?

The context is a bunch of (ascii formatted) data that's been transmitted through various channels and could conceivably be broken on a bit-level. So the first sanity check I'd like to perform is to test whether the received data still "looks like a number" (and then in the second step I'll perform a couple sanity checks on whether the number is in a valid range and such).

I am perfectly able and willing to write my own parser that looks at the individual aspects of the string step-by-step (is the string empty? Is the first character a digit? etc etc) but parsing for something like a double (that could be '-.3e-4' or such) would be a major pain.

(I tried "execute" but by itself it just throws the same errors and at least 6.3/Win it appears that the "quiet" stuff fails:

```
IDL> test = execute(s)
Type conversion error: Unable to convert given STRING to Double.
Detected at: $MAIN$             1 C:\Documents and
Settings\idl\test.pro
IDL> test = execute(s,/QuietExecution)

test = execute(s,/QuietExecution)
                          ^
Keyword parameters not allowed in call.
```

Dunno.)

Anybody have something clever at hand?

Subject: Re: Testing validity of numerical content of a string
Posted by Sven Geier on Thu, 07 Dec 2006 06:21:26 GMT
View Forum Message <> Reply to Message

Brian Larsen wrote:

> I just read you post a little closer and the numbers you have are more
> complicated but not too bad.
>
> IDL> print, stregex('hello', '[a-df-zA-DF-Z]', /boolean)
> 1
> IDL> print, stregex('3.14159', '[a-df-zA-DF-Z]', /boolean)
> 0
> IDL> print, stregex('3.14159e3', '[a-df-zA-DF-Z]', /boolean)
> 0
>
> To make it work perfectly you would need to know a little more info
> about what could be there but if's based on these regular expressions
> can tell the difference between 'hello' and '-.3e-4' which can then
> build valid_float.pro.

I kinda mostly agree with the regexp direction, but I'd recommend looking for a lot more than just "what characters are in the string". Order counts, for example: -1.e4 -1e.4 1e.-4 1-e4. all contain the same characters, but not all are well-formed doubles.

(And if the OP manages to get bit-errors that turn "3.141" into "hello", then I'd like to meet those bit-errors... ;)

I'm somewhat rusty on this, but the full regexp for a double would probably be something something like

 -?[0-9]\.?[0-9]*([de]-?[0-9]{1,2})?

i.e. zero or more spaces, then zero or one minus sign, then zero or more digits, then zero or one literal period followed by zero or more additional digits followed by zero or one instance of (the letter 'e' or 'd' followed by zero or one minus sign followed by one or two digits).

And this probably doesn't cover all of it...

I always found IDLs implementation of the regexp library to be rather slow; so this might not be practical for a large amount of data anyways.

In a pinch one might just force the conversion and catch any resulting error-messages with ON_ERROR -- that's ugly, but might be a lot easier than trying to parse strings...

- S

--
http://www.sgeier.net
My real email address does not contain any "Z"s.

Subject: Re: Testing validity of numerical content of a string
Posted by Michael Galloy on Thu, 07 Dec 2006 17:43:03 GMT
View Forum Message <> Reply to Message

Sven Geier wrote:
> In a pinch one might just force the conversion and catch any resulting
> error-messages with ON_ERROR -- that's ugly, but might be a lot easier than
> trying to parse strings...

So here's some code that does that:

```
function validate_numeric, value
  compile_opt strictarr

  on_ioerror, badValue

  f = float(value)
  return, 1B

  badValue: return, 0B
end
```

The odd thing about this is what IDL will do without complaining. IDL will do this without any error:

```
IDL> help, float('0.0hello5')
<Expression>    FLOAT     =       0.00000
```

but draws the line at:

```
IDL> help, float('hello5')
% Type conversion error: Unable to convert given STRING to Float.
% Detected at: $MAIN$
<Expression>    FLOAT     =       0.00000
```

Mike
--
www.michaelgalloy.com

Subject: Re: Testing validity of numerical content of a string

Posted by Bob[3] on Thu, 07 Dec 2006 18:08:25 GMT
View Forum Message <> Reply to Message

On Dec 7, 12:43 pm, "mgal...@gmail.com" <mgal...@gmail.com> wrote:
> The odd thing about this is what IDL will do without complaining. IDL
> will do this without any error:
>
> IDL> help, float('0.0hello5')
> <Expression> FLOAT = 0.00000
>
> but draws the line at:
>
> IDL> help, float('hello5')
> % Type conversion error: Unable to convert given STRING to Float.
> % Detected at: $MAIN$
> <Expression> FLOAT = 0.00000

Looks like Sven's regex is close to what IDL is doing. (except that
both '+' and '-' are accepted and any exponent can have any number of
digits, try e.g. help, float(' +5e00001hello')

Subject: Re: Testing validity of numerical content of a string
Posted by Brian Larsen on Thu, 07 Dec 2006 18:18:03 GMT
View Forum Message <> Reply to Message

Excellent solution. I haven't played with on_ioerror but certainly
will now.

Funny how you get stuck in a rut of how you normally do things even if
its not the best, in fact probably never the best way :)

Brian

mgalloy@gmail.com wrote:
> Sven Geier wrote:
>> In a pinch one might just force the conversion and catch any resulting
>> error-messages with ON_ERROR -- that's ugly, but might be a lot easier than
>> trying to parse strings...
>
> So here's some code that does that:
>
> function validate_numeric, value
> compile_opt strictarr
>
> on_ioerror, badValue

>
>   f = float(value)
>   return, 1B
>
>   badValue: return, 0B
>   end
>
> The odd thing about this is what IDL will do without complaining. IDL
> will do this without any error:
>
>   IDL> help, float('0.0hello5')
>   <Expression>   FLOAT   =   0.00000
>
> but draws the line at:
>
>   IDL> help, float('hello5')
>   % Type conversion error: Unable to convert given STRING to Float.
>   % Detected at: $MAIN$
>   <Expression>   FLOAT   =   0.00000
>
> Mike
> --
> www.michaelgalloy.com
