Subject: [ANN] MIDLE - Almost an Alternative to EXECUTE
Posted by SonicKenking on Thu, 24 Jul 2014 07:00:39 GMT
View Forum Message <> Reply to Message

Mini IDL Evaluator (MIDLE) evaluates simple IDL statements and most expressions without EXECUTE, i.e. virtual machine safe. It can be an alternative to EXECUTE in many cases.

GitHub repo:
https://github.com/ywangd/midle

It is currently at version 0.1.0 and can be also be downloaded at
https://github.com/ywangd/midle/archive/v0.1.0.zip

MIDLE implements its own parser and evaluates simple IDL statements and expressions without resorting to the power of `EXECUTE`. It even adds additional language features such as syntax for HASH and LIST literals, higher level array concatenation, bettering support for chaining function/method calls and subscripts.

MIDLE is however not without limitations. Some limitations are due to the limit of IDL language itself, notably output arguments and object property access (object method calls are OK). Others are deliberately set by design to meet the scope of MIDLE, notably program control constructs. Please refer to the GitHub page for details.

MIDLE requires IDL 8.0 or up (8.3 is recommended).

Here are a few examples using MIDLE (full documentation can be found at the GitHub repo page).

```
------------------------------------------------------------ --------
; Mandatory classic example
print, midle('"Hello, World!"')
midle, 'print, "Hello, World!"'

; Array of strings
midle, ['print, "STAR"', 'print, "WARS"']
; Or write them in one line
midle, 'print, "STAR" & print, "WARS"'

; Evaluate the content of given file
midle, 'filename', /file
; Passing variables
env = {num: 50}
print, midle('indgen(2,3,4, start=num)', env)

; Procedure call
midle, 'plot, indgen(50, start=100), /ynozero'
```

```
; Expressions
print, midle('-2.2 - 2 mod ((42. + 22) ^ 2 > 3 - 4.2) * 2.2 / 2.4')
print, midle('x eq 42 ? indgen(5, start=x) : indgen(5)', {x: 42})

; Assignment
print, midle('x = 42', env)
print, env.x  ; output 42
print, midle('h = Hash()', env)
print, midle('h["a"] = indgen(3,4,5)', env)
print, midle('h["a", 0, 1, 2] = 420', env)
print, (env.a)[0,1,2]  ; output 420
; List literal
print, midle('("a", "list", "literal")')

; Hash literal
print, midle('h{"x": 42, "y": 22, "description": "This is a hash literal"}')

; Higher level array concatenation:
env = {a: indgen(6,5,4,3,2), b: indgen(6,5,4,3,2, start=720)}
help, midle('[ [[[[a]]]], [[[[b]]]] ]', env)  ; concatenate on the 5th dimension

; Better support for chained function/method calls and subscripts
print, midle('list(indgen(3,4,5,6)[*,0:3:2,4,*][2,*,0,0:5:2], /extract).count()')
```

 -- --------

Comments and suggestions are welcome.

I'd like to thank Mike Galloy for his wonderful mgunit and idldoc, which I used extensively for developing MIDLE.

Cheers,
Yang