HOW TO WRITE VIM PLUGINS WITH PYTHON

Tweet

I'm not going to dive into how good or extendible Vim is. If you are reading this article, you probably know that. The thing that makes Vim so good, is the scripting environment behind it called VimL. Using this scripting language, you can write any functionality/plugin you need for Vim. Each plugin you use is written in this language. Here's the best part. You only need very little knowledge of VimL to be able to write plugins, if you know Python (or Ruby).

Thu 03 February 2011
By Dejan Noveski
In Coding.
tags: python plugin vim coding

WHAT'S A VIM PLUGIN ANYWAY

A Vim plugin is a .vim script that defines functions, mappings, syntax rules, commands that may, or may not, manipulate the windows, buffers, lines. It is a complete piece of code with some specific functionality. Usually, a plugin consists of several functions mappings command definitions and event hooks. When writing vim plugins with Python, often, everything outside the functions is written in VimL. But those are vim commands and they can be learned fast. In fact, VimL can be learned fast, but using python gives so much flexibility. Think about using urllib/httplib/simplejson for accessing some web service that helps editing in Vim. This is why most of the plugins that work with web services are usually done in VimL+Python.

ANY PREREQUISITES?

You must have vim compiled with +python support. You can check that using the command:

```
vim --version | grep +python
```

Vim package in Ubuntu and it's derivatives comes with +python support.

TO WORK - VIMMIT.VIM

What's better than starting with a simple example? This is a plugin that, when called, will retrieve the homepage of Reddit and will display it in the current buffer.

Start by opening "vimmit.vim" file (in vim). Since we are writing python code, its good to check if Vim supports Python:

```
if !has('python')
    echo "Error: Required vim compiled with +python"
    finish
endif
```

This piece is writen in VimL. It's best if we stick to VimL for things like this, mappings and event hooks. This function will check if Vim has python support or it will end the script with an error message.

We continue with the main function Reddit(). This is where we use Python and do the main functionality:

```
" Vim comments start with a double quote.
```

```
" Function definition is VimL. We can mix VimL and Python in
" function definition.
function! Reddit()

" We start the python code like the next line.

python << EOF
# the vim module contains everything we need to interface with vim from
# python. We need urllib2 for the web service consumer.
import vim, urllib2
# we need json for parsing the response
import json

# we define a timeout that we'll use in the API call. We don't want
# users to wait much.
TIMEOUT = 20
URL = "http://reddit.com/.json"

try:
    # Get the posts and parse the json response
    response = urllib2.urlopen(URL, None, TIMEOUT).read()
    json_response = json.loads(response)

    posts = json_response.get("data", "").get("children", "")

    # vim.current.buffer is the current buffer. It's list-like object.
    # each line is an item in the list. We can loop through them delete
    # them, alter them etc.
    # Here we delete all lines in the current buffer
    del vim.current.buffer[:]

    # Here we append some lines above. Aesthetics.
    vim.current.buffer[0] = 80*"-"

    for post in posts:
        # In the next few lines, we get the post details
        post_data = post.get("data", {})
        up = post_data.get("ups", 0)
        down = post_data.get("downs", 0)
        title = post_data.get("title", "NO TITLE").encode("utf-8")
        score = post_data.get("score", 0)
        permalink = post_data.get("permalink").encode("utf-8")
        url = post_data.get("url").encode("utf-8")
        comments = post_data.get("num_comments")

        # And here we append line by line to the buffer.
        # First the upvotes
        vim.current.buffer.append("↑ %s"%up)
        # Then the title and the url
        vim.current.buffer.append("    %s [%s]"%(title, url,))
        # Then the downvotes and number of comments
        vim.current.buffer.append("↓ %s    | comments: %s [%s]"%(down, comments, permalink,))
        # And last we append some "-" for visual appeal.
        vim.current.buffer.append(80*"-")
```

```
except Exception, e:
    print e

EOF
" Here the python code is closed. We can continue writing VimL or python again.
endfunction
```

Save the file, source it in vim (:source vimmit.vim) and:

```
:call Reddit()
```

Now, the way we call the function is not so elegant. So we define a command:

```
command! -nargs=0 Reddit call Reddit()
```

We define the command :Reddit to call the function. After adding this, open a new bufer and do :Reddit . Home page will be loaded in the buffer. The -nargs argument states how many arguments the command will take.

FUNCTION ARGUMENTS, EVAL AND COMMAND

Q: How does one access functional arguments?

```
function! SomeName(arg1, arg2, arg3)
    " Get the first argument by name in VimL
    let firstarg=a:arg1

    " Get the second argument by position in Viml
    let secondarg=a:1

    " Get the arguments in python

    python << EOF
    import vim

    first_argument = vim.eval("a:arg1") #or vim.eval("a:0")
    second_argument = vim.eval("a:arg2") #or vim.eval("a:1")
```

You can define a function with arbitrary number of arguments by putting "..." instead of argument names. You can access these arguments only by position, and you can mix them with named arguments (arg1, arg2, ...)

Q: How can I call Vim commands from Python?

```
vim.command("[vim-command-here]")
```

Q: How to define global variables and access them in VimL and Python?

Global vars are prefixed with g:. If you want to define one in your script, best thing to do is check if it exists and if doesn't define it and assign some default value to it:

```
if !exists("g:reddit_apicall_timeout")
    let g:reddit_apicall_timeout=40
endif
```

You can access it from python using the vim module:

```
TIMEOUT = vim.eval("g:reddit_apicall_timeout")
```

If you want to override this setting, you can write:

```
let g:reddit_apicall_timeout=60
```

in .vimrc .

ADDITIONAL NOTES

VimL is pretty easy once you try it. Remember that print works and everything you can do with python, you can do in here. Here you can find the documentation for the vim python module. Vimdoc is the possibly the only resource you will need when writing vim plugins.

You can also check this IBM developerWorks article .

Now, try to extend "vimmit.vim" so the user is able to choose a subreddit (as a first functional argument).

COMMENTS!

Fayaz Ahmed • 3 months ago

Hi,
I would like to know how to access text selected in visual mode from the python plugin.

1 Reply Share ›

José Hugo De la cruz • 7 months ago

Hi!, Thank you very much for this tutorial, I just created a very helpful Vim plugin in just one day! :D

Reply Share ›

Physicali2001 • a year ago

vim.error: string cannot contain newlines

I'm probably a newb here. But honestly, you need a bit more background. Like, vim ~/.vim/vimmit.vim. Oh, what the hell is up with those up and down arrows? I appreciate the new characters but damn, you're like "here you go" without any explanation. Overall, B, thanks man, you got me off in the right direction.

Reply Share ›

Piyush • a year ago

Awesome tutorial!! Thank you very much :)

Reply Share ›

anand jeyhar • 2 years ago

Does anybody know whether the python package 'vim' mentioned here is available in fedora or managed to install it from source on fedora?

Reply Share ›

Dejan Noveski Mod anand jeyhar • 2 years ago

Yes. It's called vim-enhanced. It should be in the default package repo.

Reply Share ›

anand jeyhar ➚ Dejan Noveski • 2 years ago

No,No am not asking about the vim rpm version.
In the python code 'import vim' is being used. I was referring to that package. i can't find it in pypi. the closest i found was vim_bridge, but it does something else.

Reply Share ›

Dejan Noveski Mod ➚ anand jeyhar • 2 years ago

Well, vim-enhanced has vim-python runtime that has that package out of the box. You don't have to install any additional package.

Reply Share ›

anand jeyhar ➚ Dejan Noveski • 2 years ago

Hmm.. Funny... when i open python interpreter and type import vim. it cannot find the module. ..Thanks for the quick responses though.

Reply Share ›

Dejan Noveski Mod ➚ anand jeyhar • 2 years ago

Vim package does not exist outside vim's python runtime. Only if you run your script in vim (source it), you can use the vim package. The package very closely tied with the current instance of Vim. It holds references to buffers, windows, lines, etc. It can't work outside vim.

Reply Share ›

CamiloPayan ➚ anand jeyhar • 2 years ago

http://vimdoc.sourceforge.net/...
I had the same question. Here are the docs I could find for the vim package (which yea, is only available from inside vim, not from your regular python interpreter, afaik)

Reply Share ›

Dejan Noveski Mod ➚ CamiloPayan • 2 years ago

Check the comment i made to the parent comment.

Reply Share ›

Dieter • 2 years ago

Hi, thanks for this nice tutorial.
I'm struggling with how to set/update a global variable from inside python.
vim.eval("let g:last_line='%s'" % vim.current.line) -> E121: Undefined variable: let
vim.eval("set g:last_line='%s'" % vim.current.line) -> E121: Undefined variable: set
vim.eval("g:last_line='%s'" % vim.current.line -> E15: Invalid expression: g:last_line='aa'

(the line the cursor is on contains 'aa', but can contain anything of course)

any ideas?
thanks!

Reply Share ›

Dejan Noveski Mod ➚ Dieter • 2 years ago

Hmm. Usually vim.eval("let g:var_name=something") does the trick for me. But, have in mind that value should be escaped - if the value has ' or " and it's not escaped, it will break.

Reply Share ›

Dieter ➚ Dejan Noveski • 2 years ago

Hmm even just putting literally:

vim.eval("let g:var_name=something")

yields the same E121: Undefined variable: let

for completeness, here's my entire code: http://pastie.org/1918338

Reply Share ›

Justin Riley ➦ Dieter • 2 years ago

using vim.command instead of vim.eval works for me

Reply Share ›

Bob/Paul . • 2 years ago

Due to long titles, I occasionally get

Error detected while processing function Reddit:
line 57:
Traceback (most recent call last):
File "<string>", line 44, in <module>
vim.error: string cannot contain newlines

This can be solved by replacing '.encode("utf-8")' with '.encode("utf-8").replace("\n", " ")'

thought I'd share.</module></string>

Reply Share ›

Jan Christoph Ebersbach • 3 years ago

Nice article, thank you. I started writing vim-orgmode in Python some time ago (http://www.github.com/jceb/vim.... I can recommend using it.

One thing you only mentioned at the side is data exchange between VimL and Python. I haven't found a way around using plain variables instead of nice return statements, yet. This is the only drawback since you always have to use a VimL wrapper around the Python code.

Reply Share ›

Dejan Noveski Mod ➦ Jan Christoph Ebersbach • 2 years ago

AFAIK, the only way you can exchange data between VimL and Python is through vim.eval .

Reply Share ›