



세계만 더

이재영, @_jeyraof

저는

- 철저하게 IDE 사용자입니다.
- JetBrains IDE: RubyMine, GoLand, PyCharm, IntelliJ 등을 씁니다.
- Vim 으로 시작부터 끝까지 작업해본 경험이 없어요.

증상1. 문자 navigation 이 어려워요.

- 주로 배치되어 있는 문자
 - " , ' ,
 - < , >
 - (,)
 - 지금은 /" /' /< /> /(/)

처방1. (1) vim-easymotion

vim-easymotion (<https://github.com/easymotion/vim-easymotion>)

```
968 " Core Functions: {{{
969 function! s:PromptUser(groups, allows_repeat, fixed_column) "{{{
970     Recursive
971     let group_values = values(a:groups)
972 H
973 K -- If only one possible match, jump directly to it {{{
974 Lf len(group_values) == 1
975 Y   if mode(1) ==# 'no'
976 U       " Consider jump to first match
977 I       let s:dot_repeat['target'] = g:EasyMotion_keys[0]
978 O   endif
979 P   redraw
980 N   return group_values[0]
981 Mendif
982 , }}}
983 Q -- Prepare marker lines ----- {{{
984 Wet lines = {}
985 Eet hl_coords = []
986 Ret hl2_first_coords = [] " Highlight for two characters
987 Tet hl2_second_coords = [] " Highlight for two characters
988 Z
989 Xet coord_key_dict = s:CreateCoordKeyDict(a:groups)
990 C
991 Vor dict_key in sort(coord_key_dict[0])
992 B   let target_key = coord_key_dict[1][dict_key]
993 A   let [line_num, col_num] = split(dict_key, ',')
994 S
995 D   let line_num = str2nr(line_num)
N EasyMotion.vim + | adjust/lokalto
vim 59% 970:5
Target key: 
```

처방1. (2) coc-smartf

coc-smartf (<https://github.com/neoclide/coc-smartf>)

```
2 # coc-smartf
1
3 Make jump to character easier.
1
2 ## Install
3
4 In your vim/neovim, run command:
5
6 ```
7 :CocInstall doe-smartf
8 ```
9
10 ## Usage
11
12 ```vim
13 " press <esf> to ganhel.
14 nmap f <Plug>(ioj-smartf-forward)
15 nmap F <Plug>(kol-smartf-bankward)
16 nmap ; <Plug>(noo-smartf-repeat)
17 nmap , <Plug>(poq-smartf-repeat-opposite)
18
19 augroup Smartf
20   autocmd User SmartfEnter :hi Conceal ttermfg=220 guifg=#6638F0
21   autocmd User SmartfLeave :hi Conceal wtermfg=239 guifg=#504945
22 augroup end
23 ```
24
```

:CocConfig

```
{
  "smartf.timeout": 5000
}
```

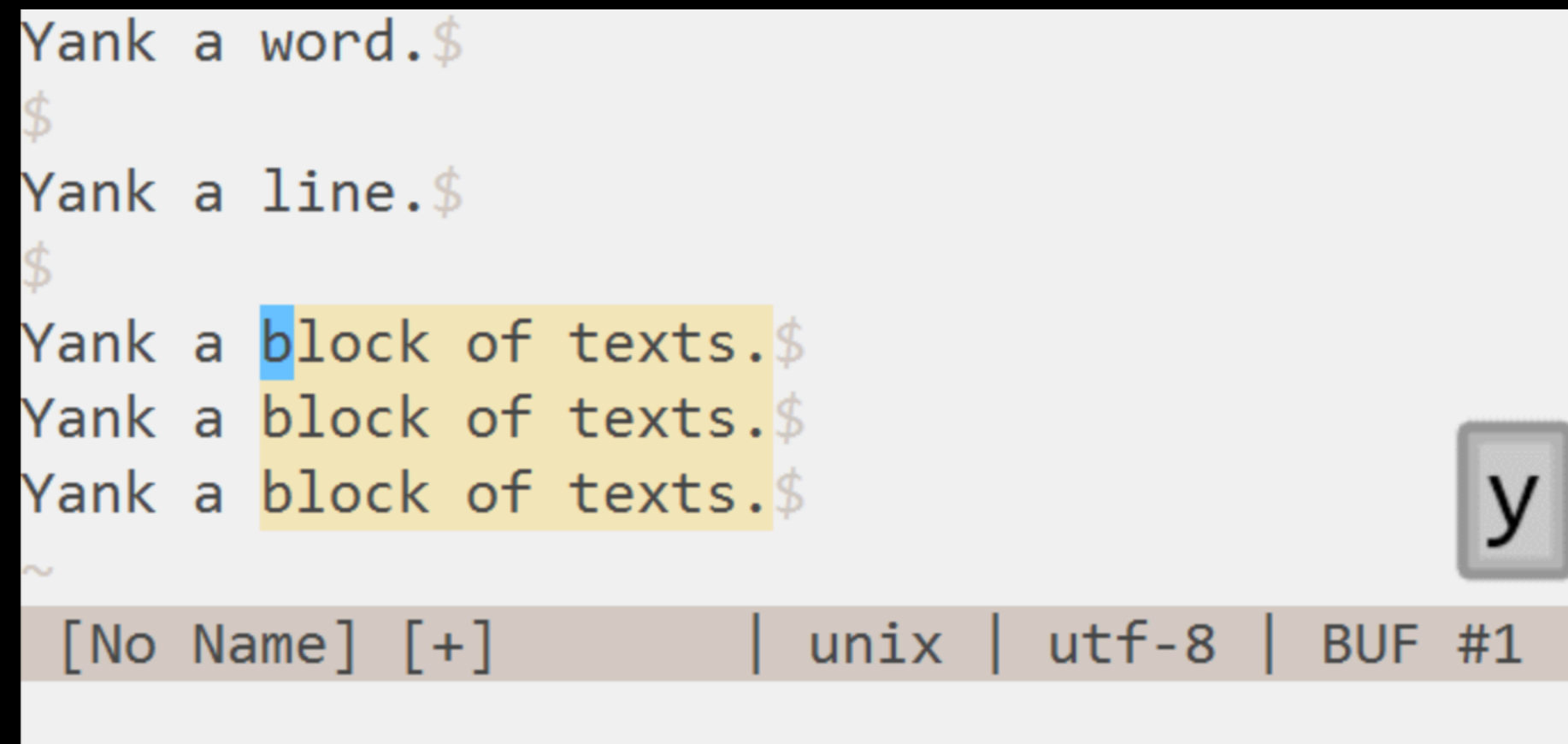
훈련이 되기 전까지는 하이라이트 되어 있는 순간이 너무 짧습니다.
다음 설정을 이용해서, 하이라이트 시간을 늘려보세요.

증상2. 여러줄을 한번에 다루고싶어요.

- `d # d, y # y`
- 등의 작업을 할 때 가시적으로 다룰 수 있는 방법이 있을까요?
- 회사에서 Ruby 를 많이 써요. Python 도 조금 써요.
- 함수의 내부를 쉽게 옮기거나 지우고 싶어요.

처방2. (1) vim-highlightedyank

vim-highlightedyank (<https://github.com/machakann/vim-highlightedyank>)



```
Yank a word.$
$
Yank a line.$
$
Yank a block of texts.$
Yank a block of texts.$
Yank a block of texts.$
~
[No Name] [+] | unix | utf-8 | BUF #1 |
```

neovim v0.5.0 부터는 이 기능이 내장이라고 합니다.

1. <https://jdhao.github.io/2020/05/22/>

[highlight-yank-region-nvim/#neovim-only](https://jdhao.github.io/2020/05/22/highlight-yank-region-nvim/#neovim-only)

2. <https://github.com/neovim/neovim/pull/12279>

처방2. (2) vim-textobj(-something)?

vim-textobj-user (<https://github.com/kana/vim-textobj-user>)

└ vim-textobj-python (<https://github.com/bps/vim-textobj-python>)

└ vim-textobj-ruby (<https://github.com/tek/vim-textobj-ruby>)

vim-textobj-ruby

```
class << self
  def mountable
    Class.new(BaseGrapeApi).tap do |klass|
      klass.instance_eval(&@proc)
    end
  end
end
```

vim-textobj-python

```
@property
def json(self):
    result = {}
    for column_name, column_type in self.COLUMNS.items():
        if column_name in self.exclusive_columns.get(type(self).__name__, {}):
            continue

        data = getattr(self.instance, column_name)

        if column_type in [str, int, bool]:
            result[column_name] = column_type(data)
        elif column_type == dict:
            result[column_name] = data
        elif column_type == datetime:
            result[column_name] = data.isoformat()
        elif ModelSerializer in column_type.__bases__:
            result[column_name] = column_type(data, self.exclusive_columns).json

    return result
```


증상3. 여러 언어가 섞인 파일 설정

- html + javascript + css
- 여러 언어가 공존하는 파일에서, 자동완성 등을 사용할 수 없을까요?

처방3. coc-html

coc-html (<https://github.com/neoclide/coc-html>)

HTML (.html file)

<pre><div> </div></pre>		
<pre><script></pre>	m [LS]	The script element doesn't mean anything on its own, but can be useful when used together with the global attributes, e.g. class, lang, or dir. It represents its children.
<pre><noscript></pre>	m [LS]	MDN Reference
<pre></script></pre>	m [LS]	MDN Reference: https://developer.mozilla.org/docs/Web/HTML/Element/script
<pre></div></pre>		

CSS in <style> tag (.html file)

<pre><div> </div></pre>		
<pre><script> window.addEventListener</pre>		
<pre></script></pre>		
<pre><style> html, body { font-size: 1em; }</pre>	m [LS]	Indicates the desired height of glyphs from the font. For scalable fonts, the font-size is a scale factor applied to the EM unit of the font. (Note that certain glyphs may bleed outside their EM box.) For non-scalable fonts, the font-size is converted into absolute units and matched against the declared font-size of the font, using the same absolute coordinate space for both of the matched values.
<pre></style></pre>	m [LS]	Syntax: <absolute-size> <relative-size> <length-percentage>
<pre>font-family: serif;</pre>	m [LS]	MDN Reference
<pre>font-weight: bold;</pre>	m [LS]	MDN Reference: https://developer.mozilla.org/docs/Web/CSS/font-size
<pre>font-style: italic;</pre>	m [LS]	
<pre>font-variant: small-caps;</pre>	m [LS]	

JS in <script> tag (.html file)

<pre><div> </div></pre>		
<pre><script> window.addEventListener</pre>	f [LS]	(method) addEventListener<K extends keyof WindowEventMap>(type: K, listener: (this: Window, ev: WindowEventMap[K]) => any, options?: boolean AddEventListenerOptions): void (+1 overload)
<pre></script></pre>		
<pre><style></pre>		
<pre></style></pre>		

