

# EEIC Vim Meetup

2022年2月21日

by [matsui54](#)

# 目次

- 自己紹介
- VimとNeovim
- プラグインの紹介
- なぜVSCodeではなく、Vimを使うのか

# 自己紹介

- 松井晴輝(GitHub ハンドルネーム: [matsui54](#))
- Vim歴: 2年弱
- 使っているエディタ: Neovim
- Vimの設定ファイルの行数:  
2400以上（コメント・空白行除く）
- Vim関係の活動:
  - 自作プラグイン開発
  - プラグインへのプルリクエストなど
  - 本体にも関わっていききたい...



## Vimに抱くイメージ

- 古い
- 不便
- ダサい
- VSCodeでよくね

⇒ 今日はそのイメージを変えたい...!



# VimとNeovim

## Vim

- 互換性重視
- どの環境でも入れやすい
- 一応新機能も追加されているが、なんか微妙...



## Neovim

- モダンな機能がどんどん追加されている  
ex. LSP (VSCoideのような言語機能), TreeSitter  
(よりよいシンタックスハイライト)
- Better defaults
- Lua (Vim scriptよりも数十倍速い) という言語で拡張可能 (もちろんVim scriptも動く)

特に理由がなければ、Neovimを使うことをおすすめします。



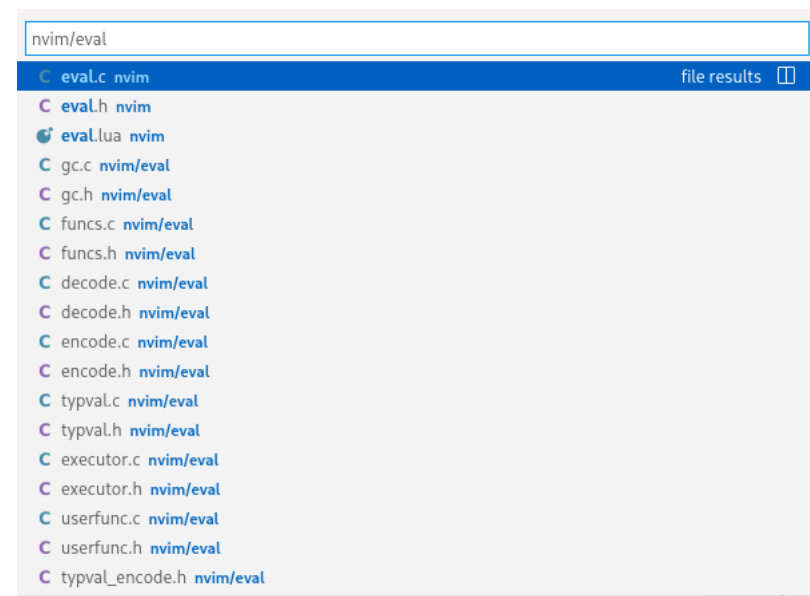
# Pluginの紹介

# ファイル検索

- ファジーファインダー
  - [denite.nvim](#) ddu.vimに開発は移行。設定は難しい。
  - [fzf.vim](#) 一番有名。
  - [Telescope.nvim](#) Neovim専用

Vimにはファジーファインダーがたくさんあるので、興味のあるかたは[こちらの記事](#)をどうぞ。

- ファイラー
  - [defx.nvim](#)
  - [fern.vim](#) 初心者におすすめ。





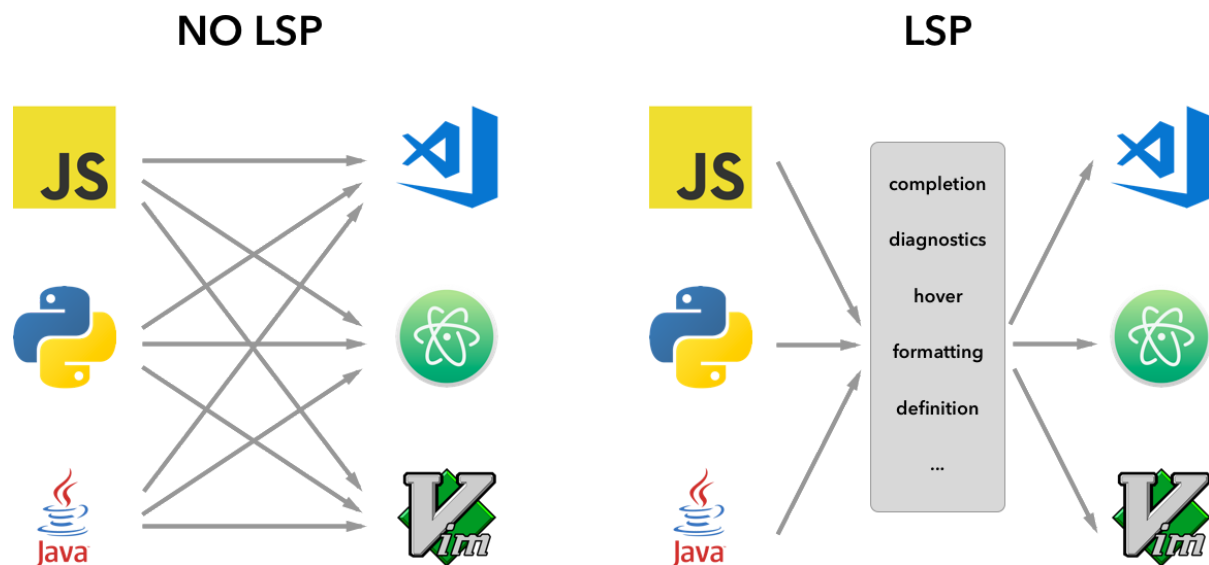
**VimでIDE的な機能を実現する**

# Language Server Protocol (LSP)

昔は、便利な編集機能（定義ジャンプや関数や変数の補完）は言語ごとに実装され、そのツールごとにエディタが対応する必要があった。

LSPはこれらの便利機能をプロトコルとして定義し、エディタはLSPの仕様を実装すればどの言語のLanguage Serverの機能も使えるようになった。

⇒ Vim/Emacs等でもLSP クライアントを入れればVSCodeの編集機能を使えるように!



## VimのLSPクライアント

- [vim-lsp](#) (Neovimではない) Vimでも動く。
- Neovim builtin LSP: Neovimの組み込みLSPクライアント。動作が速いがLuaで設定する必要あり。
- [coc.nvim](#) "Make your Vim/Neovim as smart as VSCode."がモットー。  
初心者におすすめ。自動補完機能も持っている、オールインワンプラグイン。  
Nodejsを入れる必要あり。

## LSPの機能とデモ

- 診断
- 定義ジャンプ
- 補完（引数・補完候補のドキュメントの表示）
- シンボルー覧
- 関数のヒントの表示（ホバー）
- コードの整形

## 自動補完

- [ddc.vim](#) 最強のカスタマイズ性。Denoを入れる必要あり。
- [coc.nvim](#) 初心者におすすめ。
- [nvim-cmp](#) Neovimが使うならおすすめ。

```
#include <stdio.h>
#include <string.h>

int foo(int a, int b) { return a + b; }

int main(int argc, char **argv) {
    int x = foo(10, 20);
    strncmp
    strncmp(const char *, const char *, unsigned long) f Function [lsp] int
} •strncasecmp(const char *, const char *, unsigned long) f Function [lsp] int
  •strncasecmp_l(const char *__s1, cons...__n, locale_t __loc) f Function [lsp] int
  stance [D]
  stanch [D]
  stench [D]
  sternum [D]
  stream [D]
  strict [D]
```

```
int
From <string.h>
Compare N characters of S1 and S2.
```

## ddc.vim

- source、filterの分離という考え方
  - sourceの例
    - buffer、snippet、辞書、file
  - filterの例
    - fuzzy filter ("hf"で、"hoge\_foo"にマッチする)
    - 編集距離順にソートするfilter

## スニペット

- [vim-vsniip](#) スニペット定義はVSCoode方式。ミニマルで扱いやすい。
- [ultisnips](#) VSCoodeよりもはるかに高機能。

```
// Cのfor文の例
for (${1:size_t} ${2:i} = ${3:0}; $2 < ${4:length}; $2++) {
    $0
}
```

```
% texでよく使うやつ
図\ref{$1}
```

# Git操作

- [gina.vim](#) Git操作全般。ターミナルに戻らずに直感的に操作できる。
- [git-messenger.vim](#) 行ごとにコミット情報が出せる。

```

// write an operator to being executed we return virtual_op, because
// Visual_active has already been reset, thus we can't check for "block"
// being used.
if (virtual_op != kNone) {
    return
}
return c
}

History: #0
Commit: 44cb491f6e2dd0b1aafa9ab83fe5f13cfbb716df
Author: Jan Edmund Lazo <janedmundlazo@hotmail.com> == Ctrl_V)
Author Date: Thu 19 Jul 2018 01:54:58 AM JST
Committer Date: Thu 02 Aug 2018 04:28:49 AM JST

/// VISUAL globals: virtual_op is TriState al to
/// NORMAL e.

int get_real_state(void)

```



## Vimの組み込み機能拡張プラグイン

- `vim-sandwich` ()や""で囲むoperatorを実現する。
- `clever-f` `f` コマンドを拡張。飛べる文字をハイライトしてくれる。
- `vim-swap` 関数の引数を一発で入れ替え。
- `undotree` 変更履歴の一覧表示。

# 見た目を改善するプラグイン

## カースキーム

- [iceberg](#): コントラスト低め
- [gruvbox-material](#)
- [solarized](#): VSCodeにもあるやつ
- [shirotekin](#): 白背景だけど見やすい
- その他たくさん...

[colorswat](#)というサイトで、カースキームを見比べることができる。



## ステータスライン

- [lightline](#)

NORMAL	[No Name]		unix		utf-8		no ft	100%	0:1
INSERT	[No Name]		unix		utf-8		no ft	100%	0:1
VISUAL	[No Name]		unix		utf-8		no ft	100%	0:1
REPLACE	[No Name]		unix		utf-8		no ft	100%	0:1

[https://raw.githubusercontent.com/wiki/itchyny/lightline.vim/image/solarized\\_light.png](https://raw.githubusercontent.com/wiki/itchyny/lightline.vim/image/solarized_light.png)

## その他便利なプラグイン

- [vim-quickrun](#) 一発でコンパイル&実行。
- [caw.vim](#) コメントアウトプラグイン。
- [calendar.vim](#) カレンダーを出せる。
- [indentLine](#) インデントを可視化する。
- [vim-tex](#) texのプレビュー等。
- [skkeleton](#) 変態的な日本語入力機構である [SKK](#) をVimで実現する。IMEを切り換える必要がない。

## なぜVSCodeではなく、Vimを使うのか

- すべてを自分の思い通りにしたい。
  - VSCodeは拡張機能や設定で、ある程度ユーザーの思い通りにすることはできる。
  - しかし、エディタにもとから組み込まれている機能については、不要な機能を削ったり、他の拡張に置きかえることはできない。
- マウスを使いたくない (Vim拡張を使ってもマウス操作を強いられることがある)

# Vimに興味を持った方へ

まずは[実践Vim](#)を読みましょう。

## おすすめ入門記事

「vim プラグイン」とかでGoogle検索上位に出てくるプラグインはすでに古くなっていることも多い...

- [上達したいVim初心者のための設定・プラグインのを見つけ方、学び方](#)  
基本的な心構えについて。記事内で言及されているvim-polyglotは現在はおすすめしません。
- [無人島に持っていく \(Neo\)vim プラグイン10選 \(TS開発環境編\)](#)
- [Neovim v0.5リリース記念 v0.5の新機能を紹介します【前編】](#)  
Neovimについて知りたい方向け。

**:qall!**