

VSCodeをEmacsキー配列で使う

以下の記事で「VSCode([Visual Studio Code](#))」をインストールした。そしてEmacsキー配列で使うようにカスタマイズしたので備忘録を残す。

🔗 [VSCodemacOS, Windows, Ubuntuで共通なエディター環境を構築（インストール編）](#)

基本キーボード配列変更

Emacs配列の場合は、CtrlキーとMetaキーが押しやすい位置にあった方が便利なので、以下の変更をする。

Windowsの場合

CtrlキーとCapsキーを入れ換えてCtrlキーが小指で押しやすい位置に移動する。

Windowsの場合、レジストリを書き換える事でキー配列は変更できるのでMicrosoftが配布していた「[Remapkey](#)」というツールを使って変更を行った。尚、このツールは現在では入手できないかもしれないが、同等機能のソフトが色々と出回っているので、どれらを利用してもよい。

macOSの場合

Emacsキー操作ではMetaキーとして使われるoptキーが親指で押しやすい位置にあった方が便利なのでcmdキーと入れ替える。🔗 [Karabiner-Elements](#)を使ってvscodeでのみキー入れ替えを行う。

以下の「vscode.json」ファイルを作成

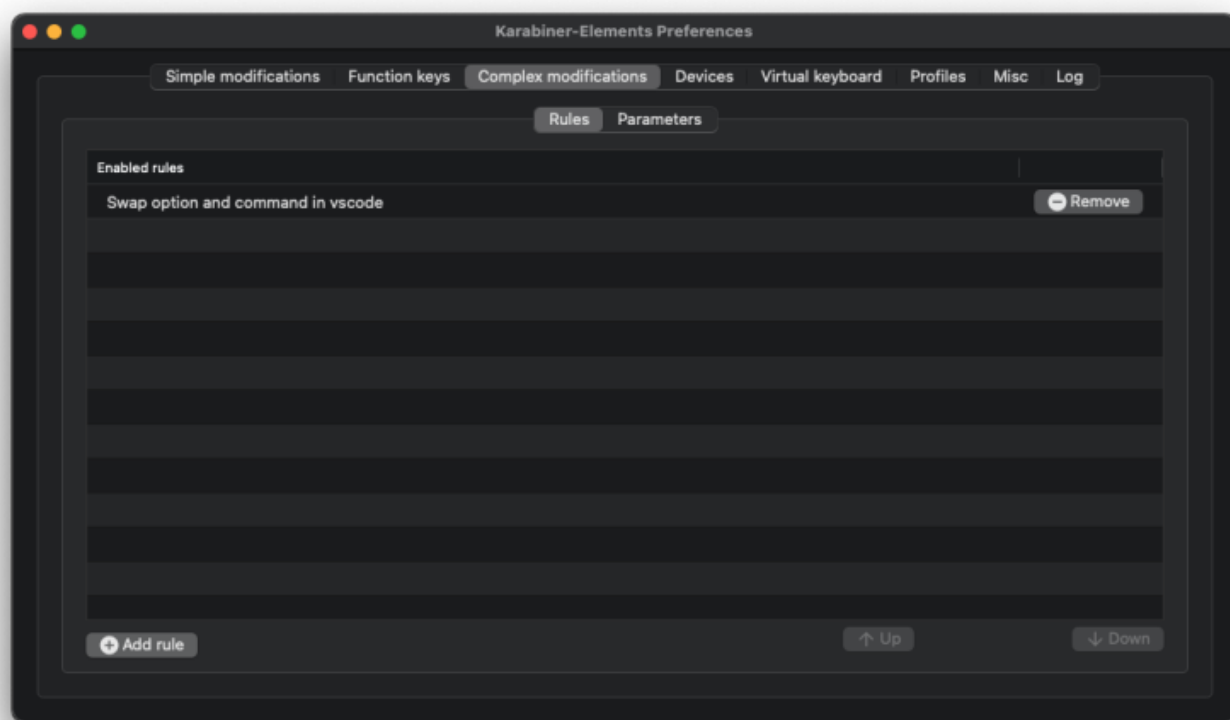
し、`~/ .config/karabiner/assets/complex_modifications/` フォルダに保存。

[vscode.json](#)

```
{
  "title": "ALT <=> CMD in VSCode",
  "rules": [
    {
      "description": "Swap option and command in vscode",
      "manipulators": [
        {
          "type": "basic",
          "from": {
            "key_code": "left_option",
            "modifiers": {
              "optional": [
                "any"
              ]
            }
          },
          "to": {
            "key_code": "command_left"
          }
        }
      ]
    }
  ]
}
```

```
    "to": [
      {
        "key_code": "left_command"
      }
    ],
    "conditions": [
      {
        "type": "frontmost_application_if",
        "bundle_identifiers": [
          "VSCode"
        ]
      }
    ]
  },
  {
    "type": "basic",
    "from": {
      "key_code": "left_command",
      "modifiers": {
        "optional": [
          "any"
        ]
      }
    },
    "to": [
      {
        "key_code": "left_option"
      }
    ],
    "conditions": [
      {
        "type": "frontmost_application_if",
        "bundle_identifiers": [
          "VSCode"
        ]
      }
    ]
  }
]
```

上記設定を、Karabinerで有効にする。



尚□Ctrlキーについては□Mac標準キーボードでは、小指の横に配置されているので変更の必要はないが、もしもWindowsキーボードなどを使っている場合などは、このソフトなどで配列変更する。

Ubuntuの場合

CtrlキーとCapsキーを入れ換えて□Ctrlキーが小指で押しやすい位置に移動する。手順は、下記記事を参照。

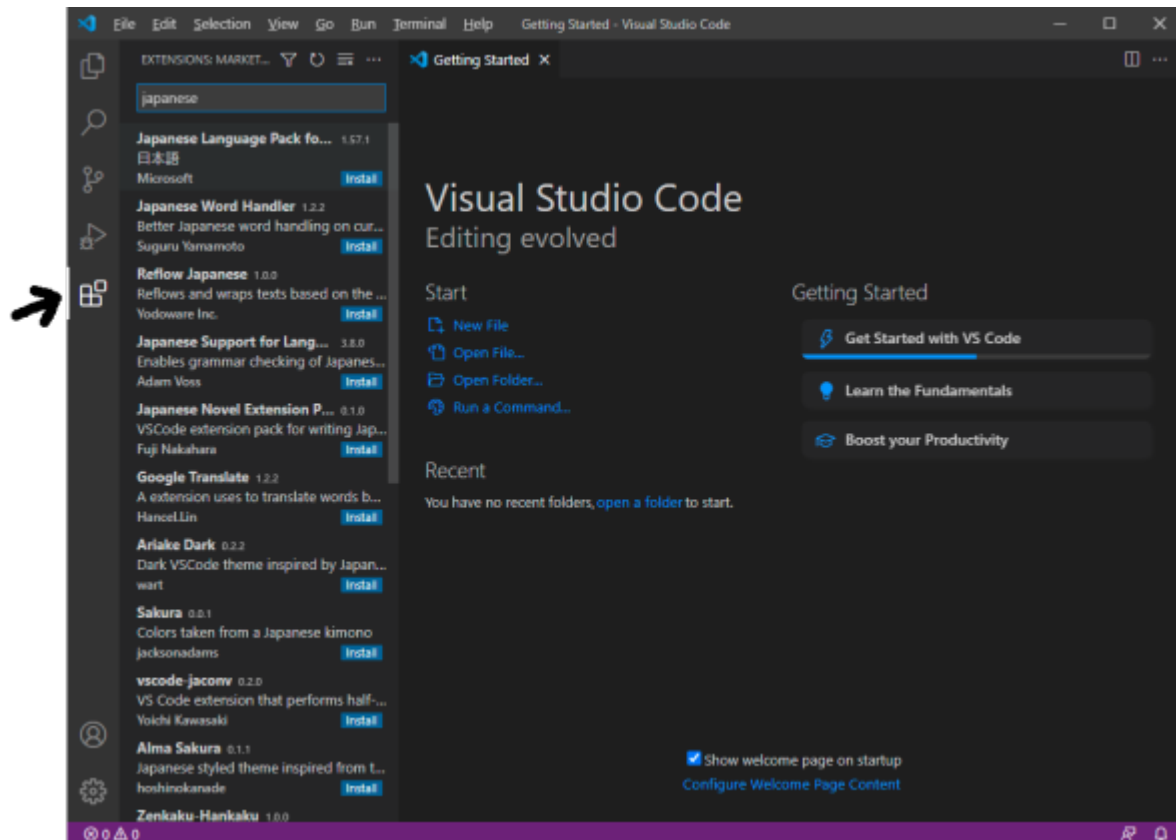
[□Ubuntu 20.04/18.04 LTS□CapsLockとControlキーを入れ替える](#)

VSCodeのキーバインド変更

拡張機能を使ってEmacsキー準拠配列にカスタマイズする。いろいろな拡張機能が公開されているが「[Awesome Emacs Keymap](#)」がお勧めでインストールする。

拡張機能のインストール

VSCodeでは、様々な拡張機能が準備されている。まずは□Emacsキー配列に変更する拡張機能をインストールする。
サイドメニューの■マークが並んだアイコンをクリック。



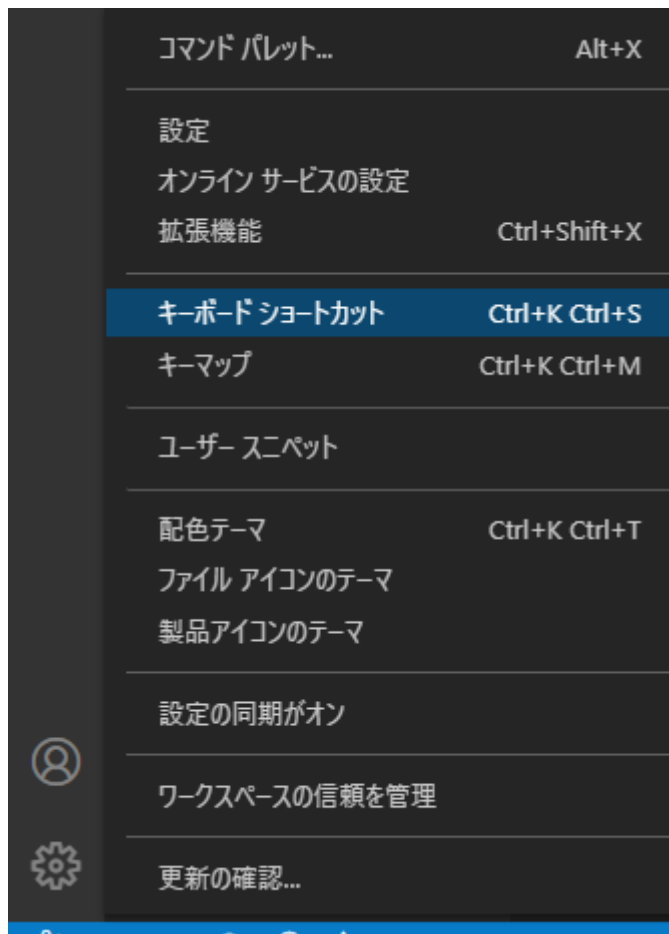
検索フォームに「Awesome Emacs Keymap」と入力する。（全部入力しなくてもマッチする一覧がでる）これで、基本的なEmacsキー操作になる。キー割り当ては、下記URLのドキュメント参照。

[Awesome Emacs Keymap](#)

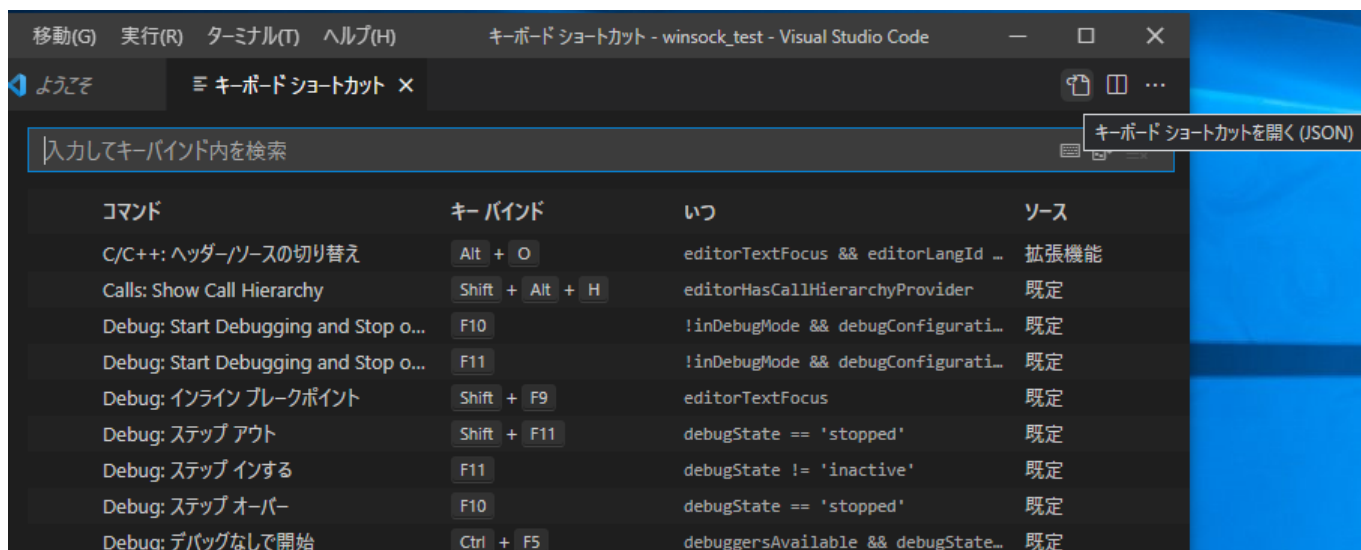
キーバインドの調整

あとは、好みのキーバインドを追加する。キーバインドは、`keybindings.json` ファイルに記述する事で設定できる。

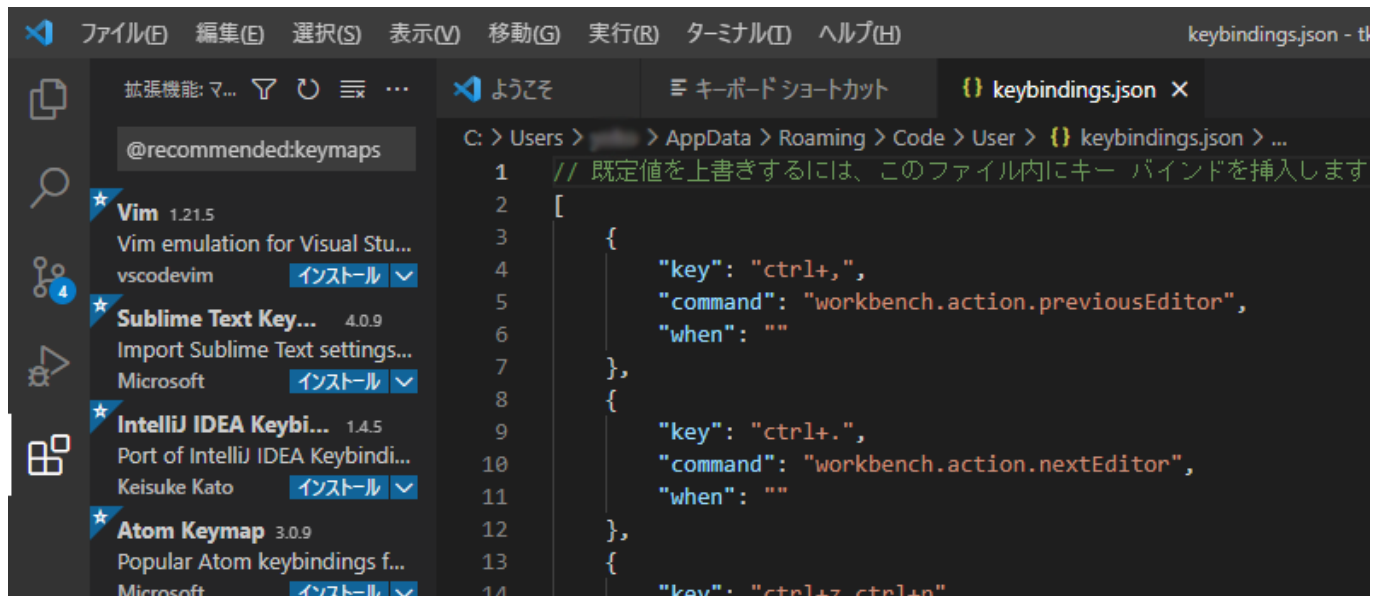
左下の歯車アイコン（設定）から「キーボードショートカット」を選択。



設定されているキーバインドの一覧が表示される。次に、右上の「キーボードショートカットを開く (JSON)」を押す。



keybindings.jsonファイルの編集画面となるので、キー割り当ての定義を記述する。



以下の定義を追加。

[keybindings.json](#)

```
[
  {
    "key": "ctrl+,",
    "command": "workbench.action.previousEditor",
    "when": ""
  },
  {
    "key": "ctrl+.",
    "command": "workbench.action.nextEditor",
    "when": ""
  },
  {
    "key": "ctrl+z ctrl+p",
    "command": "workbench.action.previousEditor",
    "when": ""
  },
  {
    "key": "ctrl+z ctrl+n",
    "command": "workbench.action.nextEditor",
    "when": ""
  },
  {
    "key": "ctrl+z ctrl+a",
    "command": "cursorTop",
    "when": "textInputFocus"
  },
  {
    "key": "ctrl+z ctrl+e",
    "command": "cursorBottom",
    "when": "textInputFocus"
  }
]
```

```
    },
    {
      "key": "ctrl+q ctrl+i",
      "command": "tab",
      "when": "editorTextFocus && !editorReadOnly &&
!editorTabMovesFocus"
    },
    {
      "key": "ctrl+q ctrl+f",
      "command": "editor.action.addSelectionToNextFindMatch",
      "when": "editorFocus"
    },
  ],
  // サイドバーのファイラーへ移動（戻る）
  {
    "key": "ctrl-z ctrl-o",
    "command": "workbench.view.explorer"
  },
]
```

以下のキー定義となる。

キー割当て	説明	コマンド
C-,	左のタブへ切り換え	workbench.action.previousEditor
C-.	右のタブへ切り換え	workbench.action.nextEditor
C-z C-p	左のタブへ切り換え	workbench.action.previousEditor
C-z C-n	右のタブへ切り換え	workbench.action.nextEditor
C-z C-a	バッファ先頭に移動	cursorTop
C-z C-e	バッファ末尾に移動	cursorBottom
C-q C-i	タブ	tab
C-z C-o	サイドバーのファイラへ移動	workbench.view.explorer

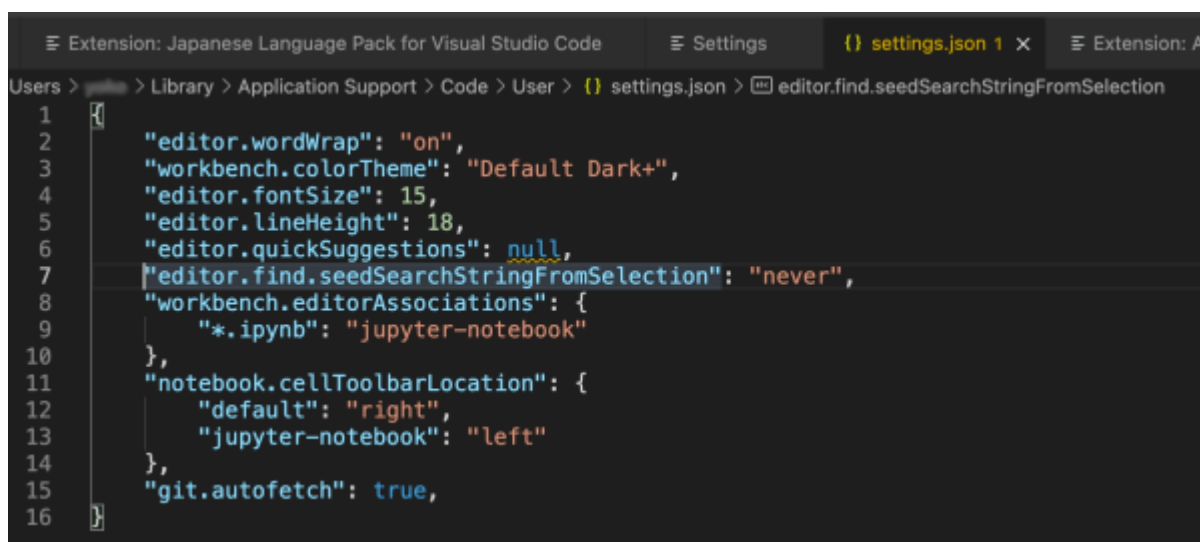
C-sキーでの検索動作

C-sキーでの検索の挙動が違っててストレスだったが、[Awesome Emacs Keymap](#)のFAQにも書かれているがEmacsと同様にするには下記の設定を追加する。

[settings.json](#)

```
"editor.find.seedSearchStringFromSelection": "never",
```

i-search (C-s) is initialized with the currently selected string and the previous search is removed. This is VSCode's design that an extension cannot control. To disable it, you should set `editor.find.seedSearchStringFromSelection` VSCode setting as “never”. It makes the find widget work similarly to Emacs.



```
1  {
2    "editor.wordWrap": "on",
3    "workbench.colorTheme": "Default Dark+",
4    "editor.fontSize": 15,
5    "editor.lineHeight": 18,
6    "editor.quickSuggestions": null,
7    "editor.find.seedSearchStringFromSelection": "never",
8    "workbench.editorAssociations": {
9      "*.ipynb": "jupyter-notebook"
10   },
11   "notebook.cellToolbarLocation": {
12     "default": "right",
13     "jupyter-notebook": "left"
14   },
15   "git.autofetch": true,
16 }
```

C-sの動きが近くなって快適になった。あとは、C-s C-wでの検索ワードの取得の挙動を同じにしたい。

検索ウィジェットの終了

[Awesome Emacs Keymap](#)のFAQにも書かれているが、ウィジェットは移動キーで閉じますので、検索ウィジェット上でのカーソル移動はできない。これは、オリジナルのEmacsの動作を意図された設計です。

尚、C-gでキーでも検索ウィジェットが終了できるが、検索開始したカーソル位置に戻ったりする仕様なので、オリジナルのEmacsとは動きが異なる。カーソル移動で抜け出すように使うようにした方がよい。

関連記事

1. [VSCodeの設定](#)
2. [VSCode macOS, Windows, Ubuntuで共通なエディター環境を構築（インストール編）](#)
3. [Emacs 25-27 macOS, Windows, Ubuntuで共通なエディター環境を構築（まとめ）](#)

参考

1. [Karabiner-Elementsで、MacのTerminalでoptionとcommandを入れ替え・自由にキーバインド変更をする例](#)
2. [快適キーボード操作のためのキーカスタマイズ Mac編～](#)
3. <https://github.com/vscode-doc-jp/Docs/blob/master/getstarted/keybindings.md>
4. [VSCodeのEmacsキーバインド拡張まとめ](#)
5. [Visual Studio Code のキーボードショートカットで発火してるコマンドを探す方法](#)
6. [VSCode のキーボードショートカットのカスタマイズ法とオススメ設定](#)
7. [VSCodeでキーバインドを設定するkeybindings.jsonが無い時の対処法](#)
8. [vscodeでtabをctrl-lで入力する](#)
9. [VSCodeでAwesome Emacs Keymap使ってるときに検索バー内でカーソル移動したい](#)
10. [【小ネタ】極力キーボードから手を離したくないライトvimmerがVSCodeに設定しているショートカット（基本的なファイル操作編）](#)

- - - - -

From:

<https://www.yokoweb.net/dokuwiki/> - 新石器Wiki

Permanent link:

<https://www.yokoweb.net/dokuwiki/software/vscode/vscode-emadcs-keybindings>

Last update: **2026/07/28 13:16**

