# 1ファイルから始めるNeovimプラグイン

こまもか

## ☆ Vimプラグイン、作ってみたいですよね

Vimを使っているなら自作Vimプラグインに憧れますよね。 そこで今回は1ファイルからプラグインを作る方法を紹介します。

今回はNeoVimでLuaを使った方法を紹介しますが、VimでもVimScriptを使って似たような方法でプラグインを作ることができます。

## ? NeoVimプラグインってどうやって作るの?

Vim/NeoVimプラグインの実態は runtimepath 上に配置されたスクリプトです。 つまり、自分が書いたスクリプトを runtimepath 上に配置すればそれがプラグインになります。

# 実際に作ってみよう

## 

Neovimにおいて lua というディレクトリは特殊であり、 require() で lua ディレクトリのファイルを読み込むことができます。

またNeovimのプラグインは慣例的に setup() という関数を呼び出してNeovimに読み込ませます。

#### 例えば自分が作ったrunit.nvimの場合はこんな感じです。

```
local function matcher(ext, executors)
        local current_file = " " .. vim.fn.expand("%:t")
        return executors[ext](current_file)
end
local function runit(executors, terminal)
        if terminal == nil then
                terminal = "terminal"
        end
        vim.cmd(":vsplit")
        vim.cmd(": " .. terminal .. " " .. matcher(vim.fn.expand("%:e"), executors))
end
local function setup(terminal, executors)
        vim.api.nvim_create_user_command("RunIt", function()
                runit(terminal, executors)
        end, {})
end
return {
        setup = setup,
```

setup() 内にコマンドの設定などを書いておくと、 setup() 関数が呼び出された際にコマンドの定義処理が実行されます。

もしコマンドを定義せずに、ユーザーに関数だけを公開させたい場合は以下のように書けばユーザーが require() して呼び出せます。

#### 関数をexport

```
-- sample.lua
local function feature()
end
local function feature2()
end
return {
    feature = feature,
    feature2 = feature2
```

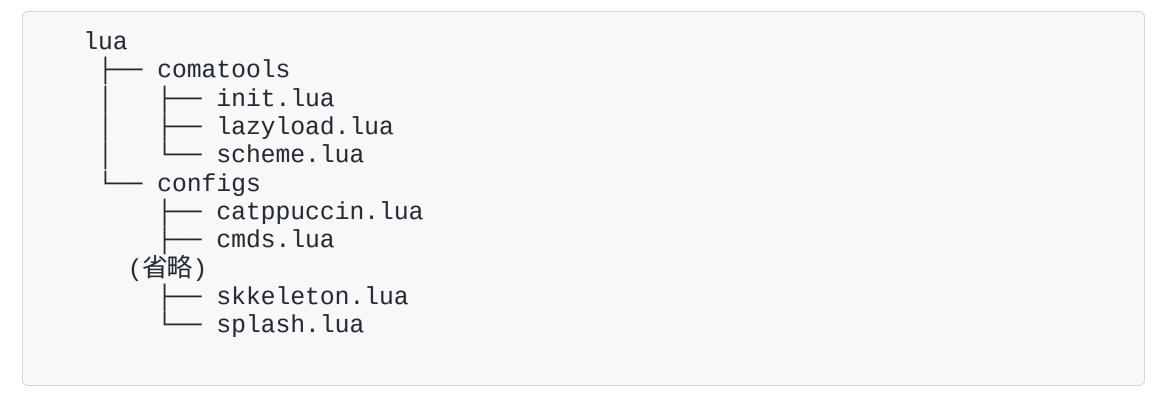
### exportされた関数をimport

```
sample = require("sample.lua")
sample.feature()
sample.feature2()
```

### ちなみに

この仕様を使うと、Vimの設定を小分けにすることが出来るのでファイル数が多くなってきたのが気になっている方はモジュール形式にして小分けにするのがオススメです。

## 自分の設定例



こんな感じで自作したプラグインもどきを comatools に、プラグイン固有の設定などは configs に配置しています。

こうする事で、 init.lua で設定ファイルを読み込むかどうかを簡単に制御すること が可能です。

#### init.luaの冒頭

```
require("configs/cmds")
require("configs/keybind")

-- require("configs/splash")

require("comatools/lazyload")
require("comatools/kit")
require("comatools/cloma")
```

## **\*\***まとめ

- NeoVimプラグインは1ファイルからでも書くことが出来る
- Luaのexportは return を使う
- 自作プラグインをプラグインマネージャでインストールすると脳汁が出るのでオススメ