



Rilkener вчера в 16:30

Neovim для full stack программиста

VIM *

Из песочницы

Tutorial



I heard he can navigate vim with hjkl and code mouse free

Я немного расскажу, как использую vim в работе full stack программиста, со своей колокольни, но для начала отвечу на некоторые вопросы, которые задают мне, когда узнают, что я использую vim при разработке проектов:

Зачем?

Хороший вопрос. Стоит для себя лично на него ответить. Зачем валандаться с vim, изучать кучу комбинаций клавиш и плагинов? Какой смысл, если есть vscode или rcharm или phpstorm? Жизнь коротка, стоит ли ее тратить на все это? Чтобы впечатлить кого-то, как на картинке в начале статьи? Да кого этим удивишь?

Я могу ошибаться, но на мой взгляд, [ответ - всегда число](#). Сколько вы за свою жизнь затратите секунд



+26



5.2K



52



днем, сколько раз вы делитесь до мышки и вернулись на клавиатуру, переводите в секунды, умножьте на 365 и на те годы, что планируете проработать программистом. Держа в голове это число

и холодный расчет, можно прикинуть, стоит ли оно того. Есть и второстепенные факторы - сколько секунд ваше IDE тратит при старте или на открытие тяжелого файла и т.д. Эти секунды – тоже ваша жизнь. Возможно, в вашем случае это будет не такое уж и большое число, ну что же, статью можно закрывать, нет смысла тратить время.

Вы так же можете сказать, что и в IDE тоже есть горячие клавиши, да, они есть, но только они мало что решают, и вы не сможете полностью вытащить мышь из своего компьютера при работе с ним, пока не включите vim-эмуляцию. А если vim эмуляция включена, то стоит ли париться с IDE, когда можно получить то же самое с помощью плагинов и голого vim-а. Вот такая палка о двух концах.

Можно ли вести серьезную разработку на vim?

Разработка не то что будет уступать IDE, а, наоборот, их опережать за счет гибкости и настройки именно под ваши нужды. Да и быстродействие vim никто не отменял. Плюс бесчисленные комбинации клавиш, которые сидят на уровне подкорки. Тут вопрос затраченных усилий, то, что вы получите в IDE из коробки сразу же, в vim придется разобраться, как все работает и какие нужны для этого плагины или самописные функции. Это как санки тащить в гору – тяжело, но зато потом можно нестись с горы как ветер. Проблема лишь в освоении и тренировки навыка, а также в том, сколько времени на этот навык уйдет.

Тут был забавный [комментарий](#) в одной из статей про vim, что человек с помощью мышки и gui обогнал своего коллегу в терминале. А вот реальная история из моей жизни. Один человек купил себе спортивную машину с ручной коробкой передач. И мой знакомый ему говорит, я тебя без проблем на своей обычной машине с автоматической коробкой обгоню. Выехали на светофор, и так оно и получилось. Человек на автомате победил, потому что знал, что за рулем спорткара - дилетант. Значит ли это, что посадив за этот спорткар профессионального гонщика, он его тоже обгонит?

Как тренировать навык работы?

Совсем новичку стоит сразу начинать работать с neovim. [Neovim](#) – это самый современный на данный момент форк vim-а. Установите [neovim](#) и введите команду

```
:Tutor
```

Пройдите все уроки, на это уйдет примерно 1.5 часа времени, но оно того стоит, если решили потратить n-усилий, чтобы освоить работу с vim. В инете можно найти на русском этот тутор.

Можно установить приложение на телефон [Vim Master](#) и проходить тесты по командам vim. Когда я начинал изучать vim, такого приложения не было, оно бы сэкономило мне тьму усилий. Когда мне скучно и рука тянется почитать новости или заняться еще какой-либо прокрастинацией, я стараюсь запустить эту приложуху. Как не странно, меня это еще и успокаивает.

Есть vim игра, но мне лично она не понравилась, хотя на вид и цвет фремастеры разные


Есть vim игра, но мне лично она не понравилась, хотя на вкус и цвет фломастеры разные.

Плюс стоит прочитать все доступные книжки по vim, которые есть на данный момент. Их не так уж и много.

Этих знаний будет достаточно, чтобы покрыть 99% проблем, оставшийся 1% покроет гугл.

О всех новинках и модных течениях, которые связаны с vim, можно читать на редите в этих четырех ветках: [1](#), [2](#), [3](#), [4](#).

Пару слов о neovim

 Возможно кто-то не знает, я кратко расскажу историю neovim и с чего все началось. Жил был программист Thiago de Arruda из Бразилии. В 2013 году он написал патч для поддержки многопоточных плагинов для vim. Но его реквест отклонили. Другой бы на его месте расстроился и плюнул, но Тиага оказался не из тех, кто легко сдаётся. Он создал форк vim, назвал его neovim. Вкатил в него асинхронную сишную библиотеку [libuv](#), стал писать тесты, начал избавляться от старого C89 кода. Его вдохновение и настойчивость увидели другие разработчики. Образовалось комьюнити. И вот через 7 лет можно сказать, что neovim - это уже стандарт дефакто в мире vi*. Короче говоря, получилось как в пословице, не было бы счастья, да несчастье помогло. В статье, если я пишу vim, то я уже имею ввиду neovim.

Плагины

С помощью плагинов можно превратить vim в IDE, которая будет настроена на ваши потребности и на ваш вкус. Полно разных туториалов или видео-уроков на этот счет. Но я бы хотел в этой статье сконцентрироваться на плагинах, которые бы ускоряли разработку. Да, есть плагины, которые занимаются внешним видом или выводят всякую там полезную информацию, или плагины, которые рассчитаны на определенный язык, например, для GO есть [vim-go](#) и т.д. Я не буду тут перечислять специфические плагины для отдельно взятых языков, тем более, что они легко гуглятся, а попробую рассказать о плагинах, которые бы подходили для всех, вне зависимости от яп, и которые, на мой взгляд, помогают быстрее писать код.

Плагин Tagbar (навигация внутри файла)



На навигацию по файлу уходит много времени, особенно, если это легаси, с кучей всего. Открыли вы файл, в котором тысяча классов и функций. Хорошо бы сразу получить их список, а так же возможность мгновенно переместиться в нужный класс или функцию. [Плагин Tagbar](#) этим и занимается. Строит отдельное окошко с правой стороны, что-то вроде дерева из классов и функций, и дает возможность по ним перемещаться.

Я добавил такие настройки себе в .vimrc (nvim/init.vim) для этого плагина:

```
" Автостарт плагина для некоторых типов файлов
autocmd VimEnter *.py,*.pl,*.js,*.php TagbarToggle
" Компактный вид у тагбара
let g:tagbar_compact = 1
" Отк. сортировка по имени у тагбара (мне хронология важнее)
let g:tagbar_sort = 0
```

Если вы переключились на окошко с tagbar и хотите развернуть или свернуть дерево с классами, то используйте клавишу `=`

Плагин NERDTree (навигация по дереву папок и файлов)



Работа плагина NERDTree - вызов окошка слева со структурой файлов и папок

Работа плагина NERDTree - вызов окошка слева со структурой файлов и папок

NerdTree плагин для навигации по файлам проекта, что-то вроде файлового менеджера. Использую, чтобы посмотреть структуру файлов и папок, а также чтобы создать файл в нужном месте. Я добавил для себя следующие настройки

```
" Показывать скрытые файлы и папки в NERDTree
let NERDTreeShowHidden = 1
" F6 для запуска и сворачивания
noremap <F6> :NERDTreeRefreshRoot<CR> :NERDTreeToggle<CR>
```

Плагин fzf - нечеткий поиск по именам файлов



Плагин fzf в действии. Ищем файл, в котором содержится слово ball

Представьте, вам нужно открыть файл, но вы не помните, в какой папке он лежит, и не помните, как он точно называется, все что вы помните, это то, что в имени файла есть слово ball и что у него расширение вроде .cpp. В жизни обычно именно так и происходит. На gif-ке выше видно, как происходит поиск и открытие нужного файла.

Плагин fzf обеспечивает просто невероятную скорость работы с файлами и папками, и с нечетким поиском. В больших проектах на поиск и открытие нужного файла уходит много времени и сил, но с

помощью этого плагина все это становится парой пустяков. Так же если у вас открыто множество вкладок (буферов) и тоже стоит проблема быстро переключиться на нужный файл, то fzf вам в помощь.

Плагин понимает .gitignore, так что он не будет искать среди картинок. Я добавил такие настройки:

```
" Если файл уже открыт в vim, то перейти к нужному буферу
let g:fzf_buffers_jump = 1
" Горячие клавиши для файлов и буферов
noremap <C-a> :Files<CR>
noremap <C-p> :Buffers<CR>
```

Сейчас появился модный [telescope.nvim](#), который делает нечто похожее, и там еще есть ряд наворотов. Если кто-то использует, напишите, пожалуйста, в комментариях, как вам?

Плагин ask.vim поиск внутри файлов



Плагин ask.vim поиск внутри файлов

Например, вы работаете над проектом и вам нужно открыть функцию, которая содержит слово get_all, но вы не помните где и в каком файле она лежит. На gif-ке сверху идет поиск и открытие файла по этому слову, потом показан поиск по слову def. Или, например, у вас есть класс MyClass и вам надо быстро понять, где он используется.

Мои кое-какие настройки: по клавише F4 ищет слово под курсором, а по shift F4 выдает команду для поиска

```
" Поиск слова под курсором, воскл. знак, чтобы не было автооткрытия файла
noremap <F4> :Ack! <word> --ignore-dir={static,logs,files}<cr>
noremap <S-F4> :Ack! --ignore-dir={static,logs,files}
```

Плагин deoplete.nvim (асинхронная автодополнялка)



Плагин deoplete.nvim - асинхронный автокомплит

Плагин [deoplete.nvim](#) - асинхронный автокомплит. Очень быстрый. Здесь список [поддерживаемых](#)

языков программирования. Раньше я использовал [coc.nvim](#) для этих целей, но уже давно перешел на [deoplete.nvim](#). Вот то, что в IDE есть из коробки, а в vim нужно немного поразбираться.

Плагин `emmet-vim`



Трудно даже описать, что делает [emmet-vim](#), но время точно сильно экономит. Если в двух словах, он превращает аббревиатуру в код.

Смотрите на gif-ку с работой плагина. Я набрал `html:5` + `ctrl y`, и получил:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body>

</body>
</html>
```

Дальше вставил список, ввел `ul>li.my-class*>span` и получил

```
<ul>
  <li class="my-class">
    <span>один</span>
  </li>
  <li class="my-class">
    <span>два</span>
  </li>
  <li class="my-class">
    <span>три</span>
  </li>
  <li class="my-class">
    <span>четыре</span>
  </li>
</ul>
```

Все это заняло у меня несколько секунд.

Плагин surround.vim



Плагин surround.vim демо

Если понаблюдать за работой программиста, то можно увидеть, что громадная часть времени уходит на то, чтобы засунуть код в скобки или в кавычки или в какой-нибудь тег. Был код в фигурных скобках, надо эти скобки заменить на квадратные скобки, был в кавычках, надо вообще эти кавычки убрать. Почему бы не сделать это максимально быстро?

Плагин [surround.vim](#) - даст супер скорость в работе с обрамляющими кавычками, скобками и тегами. На гиф-ке видно, как это все работает.

О некоторых настройках

Навигация только через hjkl

Самая жесткая и кошмарная настройка, которую надо внести на первых порах это:

```
nnoremap <Left> :echoe "Use h"<CR>
nnoremap <Right> :echoe "Use l"<CR>
nnoremap <Up> :echoe "Use k"<CR>
nnoremap <Down> :echoe "Use j"<CR>
```

Отключаем стрелочки. По началу можно возненавидеть все на свете, каждый раз рука будет тянуться к стрелочкам и каждый раз получать сообщение, что, давай, используй `hjkl` для навигации. Тут читал статью, что человек, который самостоятельно научился плавать и потом решил, чтобы ему поставили правильную технику профессиональные пловцы, будет в шоке, насколько ему неудобно, но это надо просто пережить, зато потом с правильной техникой он сможет плыть эффективнее, быстрее и дольше.

ESC как jj

До клавиши ESC тянуться достаточно далеко и не очень эргономично, так что выход из режима insert многие вешают на jj или на jk, кому как удобнее

```
imap jj <Esc>
```

P.S.

Хотел еще рассказать о других плагинах и настройках, но посмотрел и увидел, что мое графоманство довело меня до того, что статья получилась очень длинная. Так что в следующий раз напишу еще. Я не считаю себя гуру vim и есть вещи, с которыми бы мне хотелось разобраться.

Теги: [vim](#), [neovim](#), [ide](#)

Хабы: [VIM](#)

Редакторский дайджест

Присылаем лучшие статьи раз в месяц



6

Карма

26.1

Рейтинг

[@Rilkener](#)

Пользователь

Комментарии 56

ПОХОЖИЕ ПУБЛИКАЦИИ

12 октября 2017 в 16:16

Немного о VIM и IDE

+3

16K

20

78 +47

2 июня 2014 в 18:40

VIM как IDE для разработки на Python

+46

134K

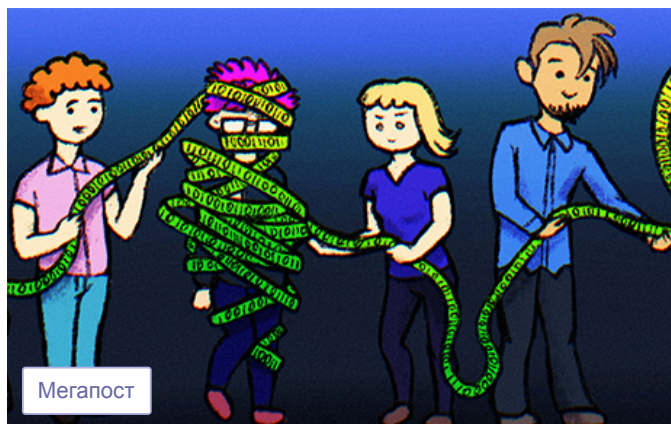
521

47 +47

12 марта 2009 в 16:16

Vim как IDE для веб-разработки, и не только.

МИНУТОЧКУ ВНИМАНИЯ



Откуда берутся Data Scientist'ы: теперь-то мы всё знаем

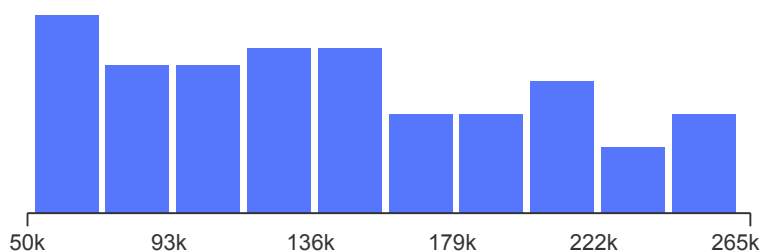


Тест Тьюринга: робот старался, но Хабр бдит

СРЕДНЯЯ ЗАРПЛАТА В IT

148 379 ₽/мес.

— средняя зарплата во всех IT-специализациях по данным из 9 843 анкет, за 2-ое пол. 2021 года. Проверьте «в рынке» ли ваша зарплата или нет!



[Проверить свою зарплату](#)

ЛУЧШИЕ ПУБЛИКАЦИИ ЗА СУТКИ

вчера в 06:05

История крови. Самая древняя проблема обратной совместимости

вчера в 06:49

Есть ли жизнь после FAANG компании или мой опыт собеседований в Северной Америке, 20+ компаний за 3 недели

♦ +27

👁 21K

🔖 83

💬 68 +68

вчера в 16:30

Neovim для full stack программиста

♦ +26

👁 5.2K

🔖 52

💬 56

вчера в 15:45

Мобильный подавитель микрофонов на Arduino. Принцип работы

♦ +24

👁 5.7K

🔖 40

💬 11 +11

вчера в 12:02

Ведьминская кривая Аньези, форму которой имеет рампа единственного российского авианосца. Почему она так названа?

♦ +18

👁 6.2K

🔖 13

💬 15 +15

Хабр



🌐 Настройка языка

О сайте

Техническая поддержка

Полная версия

Вернуться на старую версию

