Содержание

vim

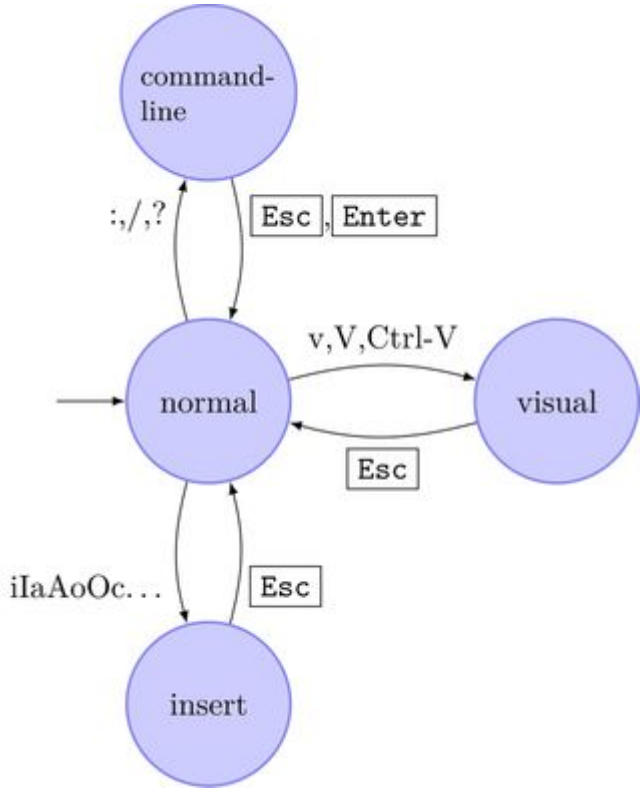
За основу взята статья с xgu.ru

Vim (**vi im**proved) — текстовый редактор, идеологическое продолжение текстового редактора vi. Отличительная особенность редактора (как и редактора vi) - поддержка режимов редактирования: поведение редактора и его реакция на нажатие клавиш определяется тем, в каком режиме он сейчас находится.

Так выглядила клавиатура создателя vi

Теория

vim modes



Буферы

Буферы - это области в оперативной памяти, куда загружается содержимое открываемых vim файлов. Каждый открытый файл считывается в отдельный буфер. Между ними можно переключаться не теряя сделанных в изменений. Содержимое буфера можно записать в файл (тот же, из которого он был прочитан, или другой).

Регистры

Регистры - это «буферы обмена» / «карманы» для хранения команд и текста.

TODO: написать о типах регистров и работе с ними.

Использование

normal mod

Переключение режимов

- `c` / `[` / `esc` - выход из режима
- `:` - **c**ommand mod
- `i` - **i**nsert mod
- v - **v**isual mod (common)
- V - **v**isual mod (lines)

- `ctrl+v` - **v**isual mod (blocks)

Быстрые действия

- `y` - скопировать
- `p` - вставить
- `x` - вырезать
- `d` - удалить
- `dd` - вырезать строку в default регистр
- `zz` - сохранить и выйти
- `ZQ` - выйти

Навигация

Внутри строки:

- `0` - переход в начало строки
- `^` - переход на первый непробельный символ
- `$` - переход в конец строки
- `f символ` - переход к первому найденному символу
- `F символ` - переход к первому найденному символу в обратном направлении
- `t символ` - перейти на один символ левее заданного (почти как f, только идёт на один символ меньше; очень удобно, когда надо «удалить до запятой, но запятую оставить»)
- `T символ` - перейти на один символ левее заданного, поиск ведется в обратном направлении
- `w` - перейти к началу следующего слова
- `e` - перейти в конец следующего слова
- `b` - перейти к началу предыдущего слова
- `ctrl+y` - сдвиг вверх, с сохранение позиции коретки
- `ctrl+e` - сдвиг вниз, с сохранением позиции коретки

По строкам:

- `gg` - перейти на первую строку
- `G` - перейти на последнюю строку
- `30G` - перейти на строку 30

По отметкам:

- ``` `` ``` - к месту, с которого был сделан последний прыжок (прыгнуть назад)
- ``` `. ``` - к месту последней правки

Прокрутка окна:

- `ctrl+y` - перемещение вверх на одну строку, с сохранением абсолютной позиции курсора
- `ctrl+e` - перемещение вниз на одну строку, с сохранением абсолютной позиции курсора
- `z-` - прокрутить окно так, что строка с курсором будет почти в самом низу
- `z.` - прокрутить окно так, что строка с курсором будет почти в центре
- `z Enter` - прокрутить окно так, что строка с курсором будет вверху

Поиск

- /слово - найти подстроку «слово»
- n - следующее вхождение
- N - предыдущее вхождение

command mod

- quit / q - выйти
- write / w - записать буфер
- pwd - текущая рабочая директория
- cd - сменить рабочую директорию
- ls - листинг буферов
- edit файл / e файл - открыть новый буфер
- buffer n / b n - сменить текущий буфер на *n*
- n - переключатся на следующий буфер
- N - переключатся на предыдущий буфер
- set опция - изменить опцию
- reg - листинг регистров
- sp - разделение экрана по горизонтали
- vsp - разделение экрана по вертикали
- help - справка
- argdo - выполнить следующие команды для всех файлов

Замена

Для замены будет использоваться такая конструкция:
`:%s/искать/заменить/g`

regex syntax

- . - соответствует любому одиночному символу, кроме символа новой строки.
- * - соответствует нулю или более вхождениям предыдущего символа или группы.
- + - соответствует одному или более вхождениям предыдущего символа или группы.
- ? - соответствует нулю или одному вхождению предыдущего символа или группы.
- ^ - соответствует началу строки.
- $ - соответствует концу строки.
- [] - определяет класс символов. Например, [abc] соответствует a, b или c.
- [^] - соответствует любому символу, не входящему в указанный класс. Например, [^abc] соответствует любому символу, кроме a, b или c.
- {n} - соответствует ровно n вхождениям предыдущего символа или группы. Например, a{3} соответствует aaa.
- {n,} - соответствует n или более вхождениям. Например, a{2,} соответствует aa, aaa и так далее.
- {n,m} - соответствует от n до m вхождений. Например, a{2,4} соответствует aa, aaa и aaaa.
- \\ - экранирует специальные символы. Например, \\. соответствует точке.
- () - группирует выражения. Например, (abc)* соответствует abc, abcabc и так далее.

- | - логическое «или». Например, abc|def соответствует либо abc, либо def.

Bash команды

Для выполнения команд в текущем pwd прямо из vim использоваться такая конструкция:
:!<команда>

insert mod

- ctrl+t - сдвинуть строку вправо
- ctrl+d - сдвинуть сроку влево
- ctrl+w - стереть слово слева
- ctrl+u - стереть всю строку слева

Настройка

TODO: написать о опциях и vimrc

Плагины

Проще всего ставить плагины при помощи плагина *vim-plug*. Его установка:

```
curl -fLo ~/.vim/autoload/plug.vim --create-dirs
https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim
```

Далее в ~/.vimrc прописываем следующее:

```
call plug#begin()

" Тут список плагинов
" Примеры:

" Сокращенная запись для GitHub; переводится как
https://github.com/junegunn/seoul256.vim.git
Plug 'junegunn/seoul256.vim'

" Любой допустимый git URL разрешен
Plug 'https://github.com/junegunn/vim-easy-align.git'

" Используя помеченный релиз; допускается подстановочный знак (требуется git 1.9.2
или выше)
Plug 'fatih/vim-go', { 'tag': '*' }

" Используя не-умолчательную ветку
Plug 'neoclide/coc.nvim', { 'branch': 'release' }
```

```
" Используйте опцию 'dir', чтобы установить плагин в не-умолчательную директорию
Plug 'junegunn/fzf', { 'dir': '~/.fzf' }

" Хук после обновления: выполните команду оболочки после установки или обновления плагина
Plug 'junegunn/fzf', { 'dir': '~/.fzf', 'do': './install --all' }

" Хук после обновления может быть лямбда-выражением
Plug 'junegunn/fzf', { 'do': { -> fzf#install() } }

" Если плагин vim находится в подкаталоге, используйте опцию 'rtp', чтобы указать его путь
Plug 'nsf/gocode', { 'rtp': 'vim' }

" Загрузка по требованию: загружается, когда выполняется указанная команда
Plug 'preservim/nerdtree', { 'on': 'NERDTreeToggle' }

" Неподдерживаемый плагин (установлен и обновлен вручную)
Plug '~/my-prototype-plugin'


call plug#end()
```