Więcej niż jeden plik i sztuczki

Bufory, okna, karty, makra i rejestry.

- Wiele plików: bufory, okna, karty
- Makra, rejestry, bloki i inne sztuczki

Wiele plików: bufory, okna, karty

Poziom: średni **Dotyczy: Vim 9.x** **Czas: ~20 min**

Administrator rzadko pracuje na jednym pliku naraz: poprawka w konfiguracji jednej usługi często wymaga zerknięcia do drugiej. Ta strona pokazuje, jak Vim trzyma kilka plików otwartych jednocześnie, oraz dwa narzędzia do samego przeglądania długich plików bez ich edycji.

Fiszka

TYLKO ODCZYT

Otwarcie kolejnego pliku bez zamykania Vima: :e

```
:e /etc/hosts
```

`:e` (edit) otwiera wskazany plik w tym samym oknie, zamiast bieżącego bufora. Poprzedni plik nie znika, zostaje w tle jako bufor, do którego można wrócić.

TYLKO ODCZYT

Lista otwartych buforów: :ls

```
:ls
```

Pokazuje wszystkie pliki otwarte w bieżącej sesji Vima, z numerem, nazwą i znacznikiem, czy mają niezapisane zmiany (`+`).

Przykład:

```
  1 %a   "/etc/fstab"                    line 1
  2      "/etc/hosts"                    line 1
```

TYLKO ODCZYT

Przełączanie między buforami: :bn

```
:bn
:bp
:b2
```

`:bn` (buffer next) przechodzi do kolejnego bufora na liście, `:bp` (buffer previous) do poprzedniego, `:b2` bezpośrednio do bufora o numerze 2 z listy `:ls`.

TYLKO ODCZYT

Podział okna poziomo: :sp

```
:sp /etc/hosts
```

Dzieli okno edytora na górę i dół, pokazując dwa pliki (albo dwa razy ten sam) jednocześnie na ekranie.

TYLKO ODCZYT

Podział okna pionowo: :vsp

```
:vsp /etc/hosts
```

To samo co `:sp`, tylko podział biegnie pionowo, na lewo i prawo. Wygodniejsze przy szerokich monitorach i krótkich liniach konfiguracji.

TYLKO ODCZYT

Przełączanie między oknami: Ctrl+w

```
Ctrl+w Ctrl+w
Ctrl+w h
Ctrl+w l
```

`Ctrl+w` naciśnięte dwa razy z rzędu przeskakuje do następnego okna. `Ctrl+w` a potem `h`, `j`, `k` albo `l` przenosi w konkretnym kierunku, gdy okien jest więcej niż dwa.

◫◫ TYLKO ODCZYT

Nowa karta: :tabnew

```
:tabnew /etc/hosts
```

Otwiera plik w zupełnie nowej karcie, widocznej jako osobna zakładka u góry ekranu. Karty w Vimie grupują okna, nie same pliki: każda karta może mieć swój własny podział na kilka okien.

◫◫ TYLKO ODCZYT

Porównanie dwóch plików: vimdiff

```
vimdiff /docker/compose/gitea/compose.yaml /docker/compose/gitea/compose.yaml.bak
```

Osobny tryb uruchamiany od razu z terminala, otwiera oba pliki obok siebie i podświetla różnice kolorem. Przydatny przed nadpisaniem pliku konfiguracyjnego kopią zapasową, żeby zobaczyć dokładnie, co się zmieni.

Przykład:

```
 services:
   gitea:
-    image: gitea/gitea:1.26
+    image: gitea/gitea:1.27
```

◫◫ **vimdiff** = Vim w trybie porównania: otwiera dwa albo więcej plików obok siebie i podświetla różnice między nimi

Podgląd bez edycji: less i more

Nie każde spojrzenie na plik wymaga otwierania go w edytorze. Do samego przeglądania, bez ryzyka przypadkowej zmiany, służą dwa mniejsze narzędzia.

Narzędzie	Przewijanie w obie strony	Szukanie	Kiedy sięgamy
`less`	tak, strzałki, spacja, `b`	`/wzorzec`, `n`, `N`	domyślny wybór do przeglądania logów i długich plików

Narzędzie	Przewijanie w obie strony	Szukanie	Kiedy sięgamy
`more`	tylko w przód, spacja, Enter	`/wzorzec`	starsze systemy, minimalne instalacje bez `less`

```
less /var/log/messages
```

Wewnątrz `less` samo `g` (bez podwojenia) skacze na początek pliku, `G` na koniec, podobnie jak `gg` i `G` w Vimie, a `q` naprawdę kończy podgląd, w odróżnieniu od Vima, gdzie `q` ma zupełnie inne znaczenie, opisane na stronie Makra, rejestry, bloki i inne sztuczki.

Z życia wzięte

> **Z ŻYCIA WZIĘTE**
>
> Tomek poprawia plik `compose.yaml` na `docker01` i chce jednocześnie zerknąć do pliku `.env` obok, żeby sprawdzić nazwę zmiennej. Zamiast zamykać jeden plik i otwierać drugi, wpisuje `:vsp .env`, dostaje oba pliki obok siebie i przepisuje nazwę zmiennej bez pomyłki.

Jak to działa

> **Pojęcie: bufor, okno, karta.** Bufor to zawartość jednego otwartego pliku w pamięci Vima, niezależnie od tego, czy jest akurat widoczny na ekranie. Okno to widoczny na ekranie fragment pokazujący jeden bufor, tego samego bufora może dotyczyć kilka okien naraz. Karta grupuje jedno albo więcej okien pod jedną zakładką u góry ekranu. Rozróżnienie ma znaczenie przy komendach z części Pierwsze kroki: `:q` zamyka bieżące okno, `:qa!` wszystkie okna wszystkich kart w całej sesji.

Częste błędy

Objaw	Przyczyna	Co robimy
`:e innyPlik` odmawia otwarcia, komunikat o niezapisanych zmianach	bieżący bufor ma niezapisane zmiany, Vim nie chce ich porzucić po cichu	zapisujemy `:w` przed `:e`, albo wymuszamy `:e!` świadomie porzucając zmiany

Objaw	Przyczyna	Co robimy
Nie wiemy, ile plików jest otwartych i który ma niezapisane zmiany	żaden pasek Vima domyślnie tego nie pokazuje na pierwszy rzut oka	`:ls`, szukamy znacznika `+` przy nazwie
Naciśnięcie `q` w Vimie nie kończy podglądu jak w `less`	`q` w Vimie zaczyna nagrywanie makra, nie jest komendą wyjścia	wyjście z Vima zawsze przez `:q`, `:wq` albo `:q!`, patrz Pierwsze kroki: otwarcie, zapis, wyjście

□□ Ćwiczenia

1. Pracujemy nad `compose.yaml` i musimy jednocześnie widzieć plik `.env` obok, bez zamykania pierwszego. Jakiej komendy użyjemy?

Odpowiedź

`:vsp .env` (albo `:sp .env`, jeśli wolimy podział poziomy zamiast pionowego). Otwiera drugi plik w nowym oknie tej samej sesji, oba widoczne naraz.

1. Dlaczego `q` w `less` kończy podgląd, a `q` w Vimie nie wychodzi z edytora?

Odpowiedź

To dwa różne programy z różnym znaczeniem tej samej litery. `less` to prosty program do przeglądania, gdzie `q` (quit) jest jego jedyną komendą wyjścia. Vim ma znacznie bogatszy zestaw komend w trybie normalnym, i `q` zarezerwował dla rozpoczęcia nagrywania makra, opisanego na stronie Makra, rejestry, bloki i inne sztuczki, a wyjście przypisał do komend z dwukropkiem.

Powiązane: Pierwsze kroki: otwarcie, zapis, wyjście · Makra, rejestry, bloki i inne sztuczki

Źródła: `man 1 vim`, `man 1 less`, `man 1 more` · `:help windows.txt`, `:help diff.txt` w samym Vimie

Makra, rejestry, bloki i inne sztuczki

Poziom: średni **Dotyczy: Vim 9.x** **Czas: ~25 min**

Powtarzalna edycja w wielu miejscach pliku nie musi oznaczać powtarzania tych samych klawiszy ręcznie. Vim nagrywa całe sekwencje jako makra, trzyma kilka fragmentów tekstu naraz w nazwanych rejestrach i edytuje prostokątne bloki kolumn jedną komendą.

Skróty użyte na tej stronie (1)

Skrót	Co znaczy
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)

Fiszka

TYLKO ODCZYT

Nagrywanie makra: q

```
qa
...
q
```

`q` naciśnięte w trybie normalnym rozpoczyna nagrywanie makra, litera zaraz po nim (tu `a`) wskazuje rejestr, do którego trafi. Wszystko, co naciśniemy potem, jest zapamiętywane, aż do ponownego `q`, które kończy nagrywanie. To jedyne znaczenie litery `q` w Vimie, w odróżnieniu od `less` i `more`, gdzie ta sama litera kończy podgląd.

TYLKO ODCZYT

Odtworzenie makra: @a

```
@a
5@a
```

`@` plus litera rejestru odtwarza zapisaną w nim sekwencję klawiszy raz. Licznik przed komendą powtarza odtworzenie: `5@a` wykonuje makro pięć razy z rzędu.

□□ TYLKO ODCZYT

Powtórzenie ostatniego makra: @@

```
@@
```

Odtwarza ponownie makro, które było uruchomione jako ostatnie, bez podawania litery rejestru jeszcze raz.

□□ TYLKO ODCZYT

Rejestry nazwane: "a

```
"ayy
"ap
```

`"a` przed operacją kasowania albo kopiowania kieruje jej wynik do rejestru `a` zamiast do bezimiennego. `"ayy` kopiuje bieżącą linię do rejestru `a`, `"ap` wkleja z tego rejestru, niezależnie od tego, co w międzyczasie trafiło do rejestru bezimiennego przez zwykłe `dd` czy `yy`.

□□ TYLKO ODCZYT

Edycja blokowa: Ctrl+v i I

```
Ctrl+v
3j
I
#
Esc
```

`Ctrl+v` wchodzi w tryb wizualny blokowy, ruch (tu `3j`) zaznacza prostokątny blok kolumn w czterech kolejnych liniach, `I` (wielkie i) wstawia tekst na początku zaznaczenia we

wszystkich zaznaczonych liniach naraz, `#` to sam wpisywany znak, `Esc` zatwierdza operację na wszystkich liniach jednocześnie.

Przykład:

```
#linia1=a
#linia2=b
#linia3=c
#linia4=d
```

□□ TYLKO ODCZYT

Komenda globalna: :g

```
:g/wzorzec/d
```

`:g` (global) wykonuje wskazaną komendę na każdej linii pasującej do wzorca w całym pliku. `:g/wzorzec/d` kasuje od razu wszystkie linie zawierające dany tekst, bez ręcznego przechodzenia po nich.

□□ TYLKO ODCZYT

Sortowanie linii: :sort

```
:sort
:10,30sort
```

`:sort` sortuje alfabetycznie wszystkie linie pliku, `:10,30sort` tylko wskazany zakres. Przydatne przy porządkowaniu list nazw hostów czy adresów IP w pliku konfiguracyjnym.

□□ **IP** = Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)

□□ TYLKO ODCZYT

Uruchomienie komendy powłoki: :!komenda

```
:!ls -la
```

Uruchamia wskazaną komendę powłoki, pokazuje jej wynik i wraca do edytora bez zmiany bufora. Różni się tym od `:r !komenda`, które wynik wstawia do tekstu w miejscu kursora.

◫◫ ZMIENIA SYSTEM

Wcięcia: >> i <<

```
>>
<<
3>>
```

`>>` zwiększa wcięcie bieżącej linii o jeden poziom, `<<` zmniejsza. Licznik przed komendą obejmuje kolejne linie: `3>>` wcina trzy linie licząc od bieżącej.

Z życia wzięte

◫◫ Z ŻYCIA WZIĘTE

Ola musi zakomentować dwadzieścia linii starych ustawień w pliku `nginx.conf` przed testem nowej konfiguracji. Zamiast wchodzić w tryb wstawiania dwadzieścia razy, nagrywa makro: `qa`, `I`, `#`, `Esc`, `j`, `q`, a potem odtwarza je dziewiętnaście razy komendą `19@a`, kończąc całą operację w kilka sekund.

Jak to działa

Pojęcie: makro. Nagrana sekwencja klawiszy przypisana do jednego rejestru, którą Vim odtwarza dokładnie tak, jakbyśmy sami naciskali te same klawisze jeszcze raz, tylko szybciej i bez pomyłek. Makro nie rozumie treści pliku, powtarza ruchy ślepo, więc działa najlepiej tam, gdzie kolejne linie mają tę samą strukturę.

Ta sama komenda komentowania wielu linii, którą stara strona autora uczyła przez zakres i wyrażenie regularne, działa też przez tryb blokowy:

Sposób	Komenda	Kiedy wygodniejszy
Zakres i podstawienie	`:10,20s/^/#/`	znamy dokładne numery linii, nie chcemy zaznaczać ręcznie

Sposób	Komenda	Kiedy wygodniejszy
Tryb blokowy	`Ctrl+v`, ruch w dół, `I`, `#`, `Esc`	widzimy zakres na ekranie, wygodniej zaznaczyć niż liczyć linie

Odkomentowanie działa lustrzanie: zakresem przez `:10,20s/^#//`, albo blokowo przez zaznaczenie kolumny ze znakami `#` w trybie `Ctrl+v` i skasowanie jej przez `x`.

Częste błędy

Objaw	Przyczyna	Co robimy
Naciśnięcie `q` w Vimie licząc, że zamknie plik jak w `less`	`q` w Vimie zaczyna nagrywanie makra, nie jest komendą wyjścia	<kbd>Esc</kbd>, potem `:q`, `:wq` albo `:q!`, patrz Pierwsze kroki: otwarcie, zapis, wyjście
Makro zaczyna nagrywać, mimo że nie chcieliśmy	przypadkowe naciśnięcie `q` i litery rejestru w trybie normalnym	`q` jeszcze raz kończy nagrywanie, potem `u` cofa niechciane zmiany wprowadzone w trakcie
`I` w trybie blokowym wstawia tekst tylko w pierwszej linii zaznaczenia	zaznaczenie zatwierdzono przed wpisaniem znaku, na przykład przez Enter zamiast bezpośredniego wpisania litery	powtarzamy sekwencję, kończąc ją od razu klawiszem <kbd>Esc</kbd> po wpisanym znaku

□□ Ćwiczenia

1. Mamy dwadzieścia linii do odkomentowania, wszystkie zaczynają się znakiem `#`. Jak zrobić to jedną komendą zakresową?

Odpowiedź

`:1,20s/^#//` (albo dopasowany zakres do faktycznych numerów linii). Wyrażenie `^#` dopasowuje znak `#` na początku linii, puste zastąpienie go usuwa.

1. Nagraliśmy makro do rejestru `b` i chcemy odtworzyć je dziesięć razy z rzędu. Jaka jest najkrótsza komenda?

Odpowiedź

`10@b`. Licznik przed `@` i literą rejestru powtarza odtworzenie makra tyle razy, ile wskazuje liczba, bez wpisywania `@b` po kolei dziesięć razy.

Powiązane: Edycja: kasowanie, kopiowanie, wklejanie, cofanie · Szukanie i zamiana · Tryby: dlaczego klawisze robią różne rzeczy

Źródła: `man 1 vim` · `:help repeat.txt`, `:help usr_10.txt`, `:help change.txt` w samym Vimie