Poruszanie się po pliku

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

W trybie normalnym kursor przesuwamy dziesiątkami różnych sposobów, każdy dopasowany do innej odległości: znak, słowo, linia, cały plik. Ta strona zbiera je wszystkie razem z mnemotechniką, skąd wzięła się każda nazwa.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera
WMS	Warehouse Management System: system zarządzania magazynem
YAML	YAML Ain't Markup Language: format plików konfiguracyjnych oparty na wcięciach

Fiszka

TYLKO ODCZYT

Ruch o jeden znak: h j k l

```
h    j    k    l
```

Lewo, dół, góra, prawo. Układ pochodzi z klawiatury terminala ADM-3A z 1976 roku, gdzie nie było osobnych strzałek kursora, tylko producent narysował groty wprost na tych czterech literach. Strzałki na klawiaturze działają w Vimie tak samo, ale `hjkl` ma przewagę: nie trzeba przenosić dłoni znad liter, więc działa szybciej i nie gubi się nawet na słabym łączu SSH, które czasem gubi kody wysyłane przez strzałki.

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera

TYLKO ODCZYT

Ruch o słowo: w b e

```
w    b    e
```

`w` (word) skacze na początek następnego słowa, `b` (back) na początek poprzedniego, `e` (end) na koniec bieżącego słowa. Trzy litery pokrywają wszystkie kierunki poruszania się słowo po słowie bez konieczności liczenia znaków.

TYLKO ODCZYT

Początek i koniec linii: 0 ^ $

```
0    ^    $
```

`0` (zero) skacze na sam początek linii, łącznie z ewentualnymi spacjami wcięcia. `^` skacze na pierwszy niepusty znak w linii, czyli pomija wcięcie, co jest przydatne w plikach z dużymi wcięciami jak YAML. `$` (koniec, jak w wyrażeniach regularnych) skacze na koniec linii.

YAML = YAML Ain't Markup Language: format plików konfiguracyjnych oparty na wcięciach

TYLKO ODCZYT

Cały plik: gg G

```
gg    G
```

`gg` (podwojone g od go) skacze na pierwszą linię pliku, `G` (wielkie, od go to end) na ostatnią. Ten sam wzorzec liter działa też w `less`, opisanym na stronie Wiele plików: bufory, okna, karty.

TYLKO ODCZYT

Konkretna linia: :42 i 42G

```
:42
42G
```

Dwukropek z numerem linii albo numer linii z wielkim `G` skaczą bezpośrednio do wskazanej linii. To odruch, gdy komunikat błędu mówi, że problem jest w konkretnej linii pliku konfiguracyjnego.

Przykład:

```
syntax error at line 42
```

TYLKO ODCZYT

Cały ekran: Ctrl+f Ctrl+b

```
Ctrl+f    Ctrl+b
```

`Ctrl+f` (forward) przewija ekran o całą stronę w przód, `Ctrl+b` (backward) w tył. Szybsze niż wielokrotne `j` przy przeglądaniu długiego pliku logów.

TYLKO ODCZYT

Pasujący nawias: %

```
%
```

Kursor postawiony na jednym z nawiasów `( )`, `[ ]` albo `{ }` i naciśnięcie `%` skacze od razu na jego parę. Przydatne przy szukaniu, gdzie kończy się blok w pliku konfiguracyjnym z zagnieżdżonymi nawiasami klamrowymi, na przykład w definicji kontenera.

TYLKO ODCZYT

Znaczniki: ma i 'a

```
ma
'a
```

`ma` (mark) zapisuje bieżącą pozycję pod literą `a`, `'a` (apostrof plus ta sama litera) wraca do niej w dowolnym momencie później, nawet po wielu innych ruchach. Nazwa dowolna litera od `a` do `z`, przydatna przy pracy w dwóch odległych miejscach tego samego dużego pliku na przemian.

Z życia wzięte

□□ Z ŻYCIA WZIĘTE

Marek z magazynu zgłasza Tomkowi błąd w skrypcie startowym WMS na `ora01`, z numerem linii w komunikacie: `syntax error at line 87`. Tomek otwiera plik, wpisuje `:87`, ląduje dokładnie tam, gdzie trzeba, i naprawia literówkę w kilka sekund zamiast przewijać cały plik strzałką w dół.

Jak to działa

Pojęcie: ruch. W terminologii Vima ruchem nazywamy każdą komendę, która przesuwa kursor: `w`, `$`, `gg`, `%` i wszystkie pozostałe z tej strony. Ruch sam w sobie tylko przemieszcza kursor, ale w połączeniu z operatorem (`d`, `c`, `y`) staje się częścią większej komendy, jak `dw` czy `d$`, co pokazuje strona Edycja: kasowanie, kopiowanie, wklejanie, cofanie.

Częste błędy

Objaw	Przyczyna	Co robimy
Strzałki nie działają albo działają z opóźnieniem przez wolne SSH	terminal gubi albo opóźnia sekwencje kodów wysyłane przez strzałki	przechodzimy na `hjkl`, które są zwykłymi literami i zawsze docierają natychmiast
`0` nie ląduje tam, gdzie zaczyna się tekst	`0` skacze na sam początek linii, łącznie z wcięciem spacjami	używamy `^`, które pomija wcięcie i ląduje na pierwszym widocznym znaku
`:42` nic nie robi albo zgłasza błąd	plik ma mniej niż 42 linie	sprawdzamy liczbę linii przez `:$` (skok na ostatnią linię) albo `gg` i `G`

□□ Ćwiczenia

1. Jesteśmy w linii 60. pliku o długości 200 linii i musimy poprawić coś w linii 15. Jakiej komendy użyjemy, żeby skoczyć tam bezpośrednio, bez wielokrotnego naciskania `k`?

Odpowiedź

`:15` albo `15G`. Obie wpisane w trybie normalnym skaczą wprost na piętnastą linię pliku, niezależnie od tego, gdzie akurat stoi kursor.

1. Dlaczego `hjkl` bywa szybsze i pewniejsze niż strzałki, mimo że robią to samo?

Odpowiedź

`hjkl` to zwykłe litery leżące w rzędzie środkowym klawiatury, więc palce nie muszą przenosić się znad liter. Strzałki wymagają sięgnięcia dłonią w bok, a na wolnym albo niestabilnym łączu SSH kody sterujące wysyłane przez strzałki bywają opóźnione albo pogubione, podczas gdy zwykłe litery docierają zawsze.

Powiązane: Pierwsze kroki: otwarcie, zapis, wyjście · Edycja: kasowanie, kopiowanie, wklejanie, cofanie · Wiele plików: bufory, okna, karty

Źródła: `man 1 vim` · `:help motion.txt` w samym Vimie · catonmat.net, „Here is why vim uses hjkl keys as arrow keys"

Revision #30
Created 2026-09-27 11:19:45 UTC by Tomasz Konieczny
Updated 2026-10-26 14:07:23 UTC by Tomasz Konieczny