Obsługa komendy sed

Co to jest `sed`?

> `sed` (stream editor) to narzędzie do przetwarzania tekstu na poziomie linii, działające linia po linii. Pozwala na automatyczną edycję plików lub strumieni tekstu bez konieczności otwierania ich w interaktywnym edytorze.

Podstawowa składnia polecenia

```
sed [opcje] 'polecenia' plik
```

W poleceniu `sed` wykonujemy polecenia edycyjne na każdej linii pliku lub strumienia.

Najważniejsze polecenia `sed` używane w przykładach

- `s/wzorzec/tekst/` – zamiana pierwszego wystąpienia wzorca w linii na podany tekst.
- `s/wzorzec/tekst/g` – zamiana wszystkich wystąpień wzorca w linii (opcja `g` – globalnie).
- `d` – usunięcie linii (np. `/^#/d` usuwa linie zaczynające się od #).
- `p` – wyświetlenie linii (często stosowane z opcją `-n`, by wyświetlić tylko wybrane linie).
- `a` – dodanie tekstu po bieżącej linii.
- `i` – wstawienie tekstu przed bieżącą linią.

Wyjaśnienie składni zamiany (substytucji) `s///` oraz znaków

- `s` – instrukcja “substitute” (zamień).
- Separator w `s///` zwykle to znak `/`, ale może być dowolny (np. `#`) aby uniknąć uciekania `/` w zamienianych wzorcach.

- `\'` – apostrofy służą do ujęcia polecenia tak, by uniknąć interpretacji przez powłokę.
- `g` – oznacza "globalnie", czyli zamień wszystkie wystąpienia.
- `d` – oznacza "delete", usuń linie pasujące do wzorca.

Obsługa znaków specjalnych - escape w `sed`

- `\\` – backslash musi być podwójnie ucieczony `\\\\` w poleceniach.
- `\.` – kropka oznacza dowolny znak, żeby faktycznie wyszukać kropkę musimy ją uciec.
- `\*`, `\^`, `\$`, `\[ \]` – także mają specjalne znaczenie, wymagają escape.
- Alternatywnie możemy zmienić separator `s/old/new/` na np. `s#old#new#`, co ułatwia pracę ze ścieżkami.

Przykłady:

Zamiana adresu IP na inny:

```
sed 's/192\.168\.1\.1/10.0.0.1/g' plik.conf
```

Zamienia wszystkie wystąpienia adresu IP 192.168.1.1 na 10.0.0.1, kropki są escape'owane.

Zamiana ścieżek za pomocą innego separatora:

```
sed 's#/var/www/html#/srv/www#g' plik.conf
```

Użycie znaku `#` zamiast `/` jako separatora pozwala uniknąć escape'ów w ścieżkach.

Komentowanie zakresu linii:

```
sed '10,20s/^/#/' plik.conf
```

Dodaje znak `#` na początku linii od 10 do 20.

Odkomentowanie zakresu linii:

```
sed '10,20s/^#//' plik.conf
```

Usuwa znak `#` z początku linii od 10 do 20.

Usuwanie pustych linii:

```
sed '/^$/d' plik.conf
```

Usuwa linie całkowicie puste.

Podwajanie backslash:

```
sed 's/\\/\\\\/g' plik.conf
```

Zamienia pojedynczy znak `\` na podwójny `\\` (używane np. przy generowaniu ścieżek w skryptach).

Zamiana z grupami przechwytywania:

```
sed 's/\(foo\) \(bar\)/\2 \1/' plik.conf
```

Zamienia wystąpienie "foo bar" na "bar foo", gdzie `\( ... \)` wskazuje grupy tekstu, a `\1`, `\2` odwołują się do nich w zamianie.

Zaawansowana zamiana adresu IP z regexp:

```
sed 's/\([0-9]\{1,3\}\.\)\{3\}[0-9]\{1,3\}/10.0.0.1/g' plik.conf
```

Zamienia dowolny adres IPv4 (4 liczby 0-999 rozdzielone kropkami) na 10.0.0.1.

Edycja pliku in-place na FreeBSD:

```
sed -i '' 's/stary/nowy/g' plik.conf
```

Podmienia wszystkie wystąpienia "stary" na "nowy" bez tworzenia kopii zapasowej, ważne jest wymienienie `''` po `-i`.

Revision #3
Created 23 November 2025 13:40:09 by Tomasz Konieczny
Updated 23 November 2025 13:50:10 by Tomasz Konieczny