06

# Para saber mais: Regex no terminal Linux e Windows

11%

ATIVIDADES
6 DE 13

DISCORD
ALURA

FÓRUM DO
CURSO

VOLTAR
PARA
DASHBOARD

CONHEÇA O
VITRINE.DEV

MODO
NOTURNO

As expressões regulares são muito úteis para realizar buscas, manipulação e validação de texto com base em padrões. No entanto, precisamos ativar o regex Engine, também conhecido como o motor da Regex, para que o padrão seja interpretado como uma expressão regular; caso contrário, será tratado apenas como uma sequência de caracteres.

Uma forma de trabalharmos com expressões regulares é através de ambientes de linha de comando, os terminais, ou como conhecemos popularmente, as famosas telas pretas.

Os ambientes Linux e o prompt de comando no Windows já possuem o motor da regex e podemos acioná-los através de comandos, que você encontra a seguir:

## Terminal Linux

1 - O comando `grep` : é amplamente utilizado para pesquisar texto em arquivos ou na saída de outros comandos. Ele permite o uso de expressões regulares para

A

67k xp

buscar padrões específicos e imprime o resultado das ocorrências no terminal.

**Exemplo:**

```
grep 'padrão_regex'
caminho_do_arquivo.txt
```

O **grep** também permite o uso de várias opções para especificar operações, com a estrutura de comando a seguir:

**Exemplo:**

```
grep opções ‘padrão_regex’
caminho_do_arquivo.txt
```

As opções são inseridas através de flags e você pode especificar os usos do comando, conheça algumas opções a seguir:

- `grep -i ou --ignore-case` : Ignora a diferenciação entre maiúsculas e minúsculas, tornando a pesquisa insensível a letras maiúsculas ou minúsculas.
- `grep -v ou --invert-match` : Inverte a correspondência, exibindo linhas que não contêm o padrão especificado.
- `grep -r ou -R ou --recursive` : Realiza uma pesquisa recursiva em diretórios e seus subdiretórios. Útil para encontrar

padrões em árvores inteiras de diretórios.

- `grep -l ou --files-with-matches` : Exibe apenas os nomes dos arquivos que contêm correspondências, em vez das próprias linhas correspondentes.
- `grep -c ou --count` : Exibe apenas o número de correspondências em cada arquivo, em vez das próprias linhas correspondentes.
- `grep -n ou --line-number` : Exibe o número da linha junto com as linhas correspondentes.
- `grep -E ou --extended-regexp` : Interpreta o padrão de pesquisa como uma expressão regular estendida (Regex) em vez de uma correspondência literal.
- `grep -f <arquivo> ou --file=<arquivo>` : Lê os padrões de pesquisa de um arquivo em vez de especificá-los diretamente na linha de comando.
- `grep -h ou --no-filename` : Suprime a exibição dos nomes dos arquivos ao imprimir as linhas correspondentes.
- `grep -P` : habilita o modo de interpretação de padrões como expressões regulares Perl (Perl-Compatible Regular Expressions ou PCRE). Isso significa que você pode usar padrões de expressões regulares mais avançados e complexos com a flag -P. As

expressões regulares Perl são mais poderosas e flexíveis do que as expressões regulares básicas usadas pelo grep por padrão. No entanto, nem todas as versões do grep suportam a opção -P, pois ela depende da biblioteca PCRE (Perl-Compatible Regular Expressions). Portanto, verifique a disponibilidade dessa opção na versão específica do grep em seu sistema.

- `man grep` : é uma opção que apresenta toda a documentação do grep no terminal.

Dica: você pode consultar o Manual GNU Grep (https://www.gnu.org/software/grep/manual/grep.html).

2 - **sed:** o `sed` é um comando usado para manipular os arquivos em texto, e podem substituir sequências de caracteres com base em padrões de expressões regulares. Confira o exemplo a seguir:

```
sed 's/padrão_regex/novo_texto/g'
caminho_do_arquivo.txt
```

3 - **awk**: O comando awk, por sua vez, ativa uma linguagem de programação de linha de comando e pode ser usada para processamento de texto. É possível aplicar padrões de expressões regulares para buscas ou manipulações.Confira o exemplo abaixo:

```
awk '/padrão_regex/ {print $1}'
caminho_do_arquivo.txt
```



## No terminal Windows

O comando `findstr` : permite pesquisar texto em arquivos ou na saída de outros comandos. Ele suporta o uso de expressões regulares básicas com a opção `/r` . Confira o exemplo:

```
findstr /r "padrão_regex" arquivo.txt
```

O Windows PowerShell oferece suporte completo a expressões regulares por meio de várias funções e operadores. Você pode usar o operador `-match` para verificar se uma string corresponde a um padrão.

**Confira o exemplo:**

```
Get-Content arquivo.txt | ForEach-Object
{ if ($_ -match "padrão_regex") { $_ } }
```

A plataforma da Microsoft possui uma excelente documentação sobre regex no Powershell (https://learn.microsoft.com/pt-br/powershell/module/microsoft.powershell.core/about/about_r view=powershell-7.3) que vale a pena você conferir.



> Dica: Lembre-se de que a sintaxe e as funcionalidades podem variar um

pouco entre as diferentes implementações de regex em sistemas Unix-like (como o Linux) e no Windows. Além disso, em ambas as plataformas, é importante compreender os fundamentos das expressões regulares para criar padrões eficazes e realizar tarefas específicas com sucesso.