Comandos Básicos

Nayara Luiza T. Moraes

Comandos Básicos
Linux

Ctrl + Alt + t = abrir o terminal

pwd = descobrir qual diretório estou

ls = retorna o conteúdo do diretório atual

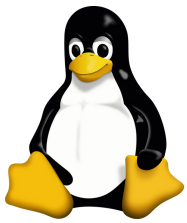
ls * = retorna o conteúdo do diretório atual e de seus subdiretórios

ls -l = detalhar, mostrar algumas informações sobre os arquivos do diretório

- o que é diretório inicia com a letra d;
- nome do usuário;
- data de modificação;
- tamanho do arquivo

OBS:

Seta para cima = volta para o comando anterior
Seta para baixo = vou para o próximo comando



Nayara Luiza

ls -la = listar todos os arquivos, inclusive os diretórios invisíveis

- diretórios invisíveis começam com ponto

cd = mudar de diretório

cd <diretorio-que-quero-ir>

cd .. = volta para o diretório anterior

cd = volta para o diretório base (home/nome-do-usuário)

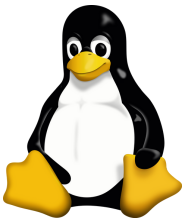
cp = copiar o conteúdo de um documento para outro

cp <nome-do-documento-que-quero-copiar> <nome-do-documento-para-onde-quero-copiar>

OBS:

. é o diretório atual

.. é o diretório anterior



cp -r = copiar um diretório

cp -r <nome-do-diretório-que-quero-copiar> <nome-do-diretório-para-onde-quero-copiar>

mv = renomear um arquivo; mover um arquivo

Renomear um arquivo

mv <nome-do-arquivo-que-quero-renomear> <novo-nome-que-quero-colocar>

Mover um arquivo

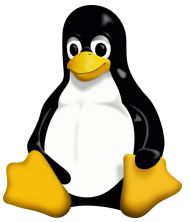
mv <nome-do-arquivo-que-quero-mover> <nome-do-diretório-para-onde-quero-mover>**/**

Mover e renomear um arquivo

mv <nome-do-arquivo-que-quero-mover> <nome-do-diretório-para-onde-quero-mover>**/**<novo-nome-do-arquivo>

OBS:

Tab = autocompleta o nome de arquivos e diretórios



Criar um arquivo

echo <mensagem> **>** <nome-do-arquivo>

1º Mostrar uma mensagem: **echo** <mensagem>
2º **>** Redirecionar a saída para um arquivo
3º Nome do arquivo

echo <mensagem> **>>** <nome-do-arquivo>

Adicionar, concatenar conteúdo

cat = imprime no terminal o conteúdo do arquivo

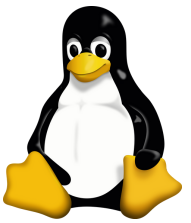
cat <nome-do-arquivo>

cat *.txt

imprime no terminal o conteúdo de todos os arquivos com a extensão txt

clear = limpar a tela do teminal

- **atalho: Ctrl + l**



man = ensina sobre o comando que quero saber mais sobre

man <nome-do-comando>

whoaim = retorna o nome do usuário que está sendo usado

zip -r = compactar um diretório

zip -r <nome-do-arquivozip-que-vou-criar**.zip**> <nome-do-diretorio-que-vou-compactar>**/**

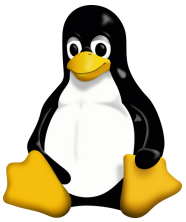
OBS: tenho que estar no diretório pai

unzip -l = conferir o que tem dentro do arquivo zip

unzip -l <nome-do-arquivozip>

unzip = para descompactar um arquivo zip

unzip <nome-do-arquivozip>



unzip -q = para descompactar o arquivo zip sem ser verborrágico

unzip -q <nome-do-arquivozip>

zip -rq = para compactar um diretório sem ser verborrágico

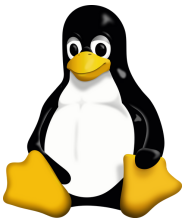
zip -rq <nome-do-arquivozip-que-vou-criar> <nome-do-diretório-que-vou-compactar>**/**

tar -cz = para compactar um diretório usando tar

tar -cz <nome-do-diretorio-que-vou-compactar> **>** <nome-do-arquivotargz-que-vou-criar**.tar.gz**>

tar -xz = para descompatar um arquivo targz

tar -xz < <nome-do-arquivotargz>



tar -czf = para compactar um diretório usando tar, sem redirecionamento

tar -czf <nome-do-arquivotargz-que-vou-criar**.tar.gz**> <nome-do-diretorio-que-vou-compactar>**/**

tar -xzf = para descompatar um arquivo targz

tar -xz < <nome-do-arquivotargz>

tar -vxzf = tornar o tar masi verborrágico e ter mais verbose

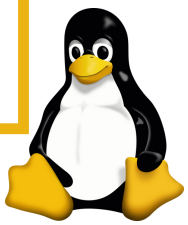
tar -vxzf <nome-do-arquivotargz>

touch = alterar a data de última modificação do arquivo

touch <nome-do-arquivo-que-quero-alterar-a-data>

head = recebe o nome do arquivo e traz as 10 primeiras linhas

head <nome-do-arquivo>



head = recebe o nome do arquivo e traz as 10 primeiras linhas

head <nome-do-arquivo>

head -n = recebe o número de linhas que quero ler e o nome do arquivo

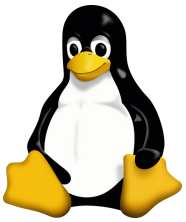
head -n <numero-de-linhas-que-quero-ler><nome-do-arquivo>

tail = recebe o nome do arquivo e traz as últimas 10 linhas

tail -n <nome-do-arquivo>

tail -n = recebe o número de linhas finais que quero ler e o nome do arquivo

tail -n <numero-de-linhas-finais-que-quero-ler><nome-do-arquivo>



less = recebe nome do arquivo e abre no terminal para navegar com as setas pelo arquivo

less <nome-do-arquivo>

vi = abrir arquivo para edição no terminal

vi <nome-do-arquivo-que-quero-editar>

i = muda do modo de navegação para o de inserção - inserir um conteúdo na posição atual

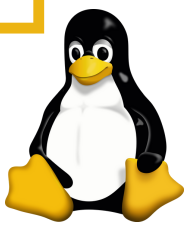
a = inserir um conteúdo na posição seguinte

A = inserir um conteúdo no final da linha

x = remover um caracter

remover mais de um caracter

<numero-de-caracteres-que-quero-apagar>x



Esc = sai do modo de edição e volta ao modo navegação

:w + enter = salvar o arquivo

:q + enter = sair do arquivo

:wq + enter = salva e sai do arquivo

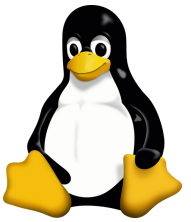
dd = remove uma linha por completo

Remover mais de uma linha

<numero-de-linhas-que-quero-remover>**dd**

:q! = ignorar as mudanças e sair sem salvar

:q! = ignorar as mudanças e sair sem salvar



G = vai para a última linha

Ir para uma determinada linha

<numero-da-linha-que-quero-ir>**G**

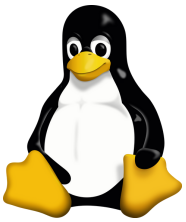
$ = vai para o final da linha atual

0 = voltar para o primeiro caracter da minha linha atual

/ = fazer uma busca

/<palavra-que-quero-buscar>

n = vou para a próxima ocorrência da palavra buscada



N = volta para a ocorrência anterior

yy = copiar uma linha

Copiar várias linhas

<numero-de-linhas-que-quero-copiar>**yy**

p= colar uma linha

Copiar várias linhas

<numero-de-linhas-que-quero-copiar>**yy**

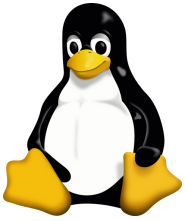
code = abrir arquivo com VS code

code <nome-do-arquivo>

xdg-open = abrir arquivo no editor de texto

xdg-open <nome-do-arquivo>

gedit = abre o editor de texto



Comandos Básicos
Git

git init = inicializar,criar o repositório git

git status = mostra o estado do repositório

git config --local user.name "Seu nome"
git congig --local user.email "Seu email"
Para informar quem é o usuário e armazenar corretamente os dados do autor de cada alteração no código
--local: para aquele repositório específico
--global: para a máquina como um todo

git config = visualizar as configurações
git config <nome-da-configuração>

git add = arquivo ou diretório passa a ser monitorado pelo git

git add <nome-do-arquivo>

git add <nome-da-pasta>**/**

git rm = remove arquivo, git deixa de monitorá-lo

git rm <nome-do-arquivo>

git commit -m = salvar alterações que foram adicionadas e uma mensagem para identificar o que foi feito

git commit -m "o-que-foi-feito"

git log = verificar histórico de commits

git log --oneline = visualizar os commits de forma resumida - cada commit ocupa apenas uma linha

git log -p = visualizar as alterações que foram realizadas em cada commit

git log --graph = mostra as linhas de desenvolvimento

git log -n2 = mostra somente os dois últimos commits

git ignore = arquivo que não queremos monitorar

.gitignore arquivo especial do git

- adicionar nesse arquivo o nome do arquivo ou pasta <nome-da-pasta>**/** que quero ignorar;
- git add .gitignore
- git commit -m "Adicionando .gitignore"

git init --bare = repositório git sem uma cópia dos arquivos, para ser utilizado como servidor

git remote = lista todos os repositórios remotos que o meu repositório conhece

git remote add = para adicionar um repositório remoto ao meu repositório

git remote add <nome-do-meu-repositório-remoto-lda-minha-máquina> <caminho-para-o-repositório-remoto-que-eu-quero-conectar>

git remote add origin <caminho-para-o-repositório-remoto-que-eu-quero-conectar>

- **origin** é o nome que o Github sugere colocar no repositório remoto principal

git remote -v = mostra o endereço do repositório

git remote rename = renomear um repositório remoto

git remote rename <nome-do-repositório-remoto-que-quero-renomear> <novo-nome>

git clone = trazer pela primeira vez todos os dados de um repositório remoto para o nosso repositório local

git clone <caminho-do-repositório-que-quero-clonar>

git push = enviar (**empurrar**) os dados de um repositório para outro

git push <para-onde> <de-onde>

git push local master

- Estou enviando os dados da minha branch master para o repositório chamado local

git pull = trazer (**puxar**) os dados de um repositório remoto para o meu repositório

git pull <nome-do-repositório-que-estou-pegando-os-dados> <para-qual-branch-quero-trazer-os-dados>

git pull local master

- Estou trazendo para a minha branch master os dados do repositório chamado local

git branch = mostra quantas branches eu tenho

git branch <nome-da-nova-branch>

git checkout = para caminhar pelas branches do meu repositorio git, para navegar nos estados do meu reositório

git checkout <nome-da-branch-que-quero-ir>

git checkout -b = atalho para criar uma nova branch e já passar para ela

git checkout -b <nome-da-branch-que-quero-adicionar>

git merge = juntar a linha secundária que estou trabalhando com a master

git merge <nome-da-branch-que-quero-juntar>

OBS: devo estar na minha branch master

git rebase = atualizar a master com commits de outra branch sem criar um commit de merge

git rebase <nome-da-branch-com-os-commits-que-quero-colocar-na-master>

**git checkout -- ** = remover alterações ainda não adicionadas

git checkout -- <nome-do-arquivo>

git restore = remover alterações ainda não adicionadas

git restore <nome-do-arquivo>

git reset HEAD = desfazer alteração que já foi marcada para commitar

git reset HEAD <nome-do-arquivo>

git restore --staged = desfazer alteração que já foi marcada para commitar

git restore --staged <nome-do-arquivo>

Desfazer um commit

git log = para copiar o hash do commit

git revert = para desfazer o commit

git rever <hash-do-commit>

OBS: **Ctrl + e** para sair da tela que vai aparecer
y para salvar e **enter**

git stash = salva as alterações em um local temporário

git stash list = lista tudo o que está salvo nesse local temporário

Pegar os dados que foram salvos no stash e trazer para meu local de trabalho para voltar nessas alterações

1º git stash list = para ver o que eu posso trazer e aplicar
2º git stash apply <nº da stash> = para aplicar
3º git stash drop <nº da stash> = para remover as alterações de lá

git stash pop = pega a última alteração da stash, remove de lá e traz para eu trabalhar nela

git checkout = fazer com que meu código fique no estado em que ele estava naquele commit

git checkout <hash>

Se eu quiser fazer alterações nesse ponto

1º criar uma nova branch

git checkout -b <nome-da-nova-branch>

git switch -c <nome-da-nova-branch>

Como gerar uma versão pronta, que já está "no ponto" de ser entregue

git checkout <nome-da-brach>

Como ver o que foi inserido em cada commit, a diferença de um commit para o outro, ver todas as modificações de uma só vez

git diff

Compara as alterações entre dois commits

git diff <hash1>..<hash2>

Compara as alterações entre duas branches

git diff <branch1>..<branch2>

Definir um commit como um marco de versão ou um marco qualquer que eu quiser

git tag -a <nome-da-tag> **-m** "mensagem"

git tag = mostra todas as tags disponíveis