



Criando e excluindo arquivos e diretórios – Linux

O comando de **“touch xxx”** quando usado em **um arquivo existente** apenas muda o **timestamp dele** para o momento que o comando foi usado, quando usado em um **arquivo que não existe** ele simplesmente **cria um novo arquivo vazio**.

```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls
'Linux - Ubuntu'
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ touch newFile newFile2 newFile3
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls
'Linux - Ubuntu'  newFile  newFile2  newFile3
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```

“mkdir xxx” cria diretórios:

```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir dir1 dir2 dir 3
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ touch dir1/arquivo
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls dir1
arquivo
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```

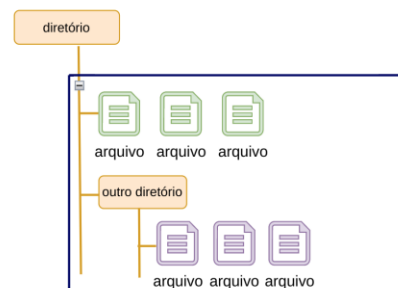
“rmdir xxx” remove **SOMENTE** diretórios que estão vazios:

```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ rmdir dir1 dir2 dir 3
rmdir: falhou em remover 'dir1': Diretório não vazio
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```

O **“rmdir xxx”** só funciona com deleção específica de um diretório que está vazio, então ele é meio inútil. Por isso o **comando mais utilizado para excluir qualquer arquivo/diretório no linux** é o **“rm xxx”**.

Ainda assim o **rm sozinho não é capaz de deletar um diretório**, ele **precisa ser usado em conjunto com a tag -R “rm -R xxxx”**, que recursivamente vai deletar todo o **diretório** e quem está **“abaixo” dele**:

```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
arquivo arquivo2 arquivo3 outrodir
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls dir1
dir1:
arquivo arquivo2 arquivo3 outrodir
dir1/outrodir:
arquivo arquivo1 arquivo2
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ rm -R dir1
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls dir1
ls: não foi possível acessar 'dir1': Arquivo ou diretório inexistente
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```



Da pra combinar o comando de “rm” com outras tags, com “-i” ele te pergunta se você quer mesmo deletar o arquivo/diretório, com “-f” você força a remoção e é isso aí, o “-f” inclusive ignora a inexistência de arquivos, por isso você nem vai ver um retorno de possível erro.

O “-R” pode ser usado em conjunto para aplicar para todos os arquivos/diretórios recursivamente: “rm xxxx -fR”, “rm xxx -iR” etc....

Outra tag bem útil é a “-v”, basicamente falando ela te retorna uma **espécie de log** sobre o que você fez com o determinado comando passado:

```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir dir1
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ touch dir1/arquivo1 dir1/arquivo2 dir1/arquivo3
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls dir1
arquivo1 arquivo2 arquivo3
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ rm -vR dir1
removido 'dir1/arquivo1'
removido 'dir1/arquivo2'
removido 'dir1/arquivo3'
foi removido o diretório: 'dir1'
```



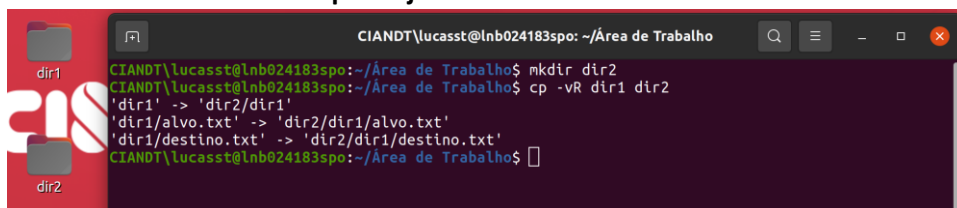
Copiando arquivos – Linux

Você pode copiar arquivos (incluindo seu conteúdo) usando o comando “cp xxx yyy”. Inclusive o comando serve também para **atualizar o conteúdo de uma cópia já criada** caso o arquivo original seja alterado:



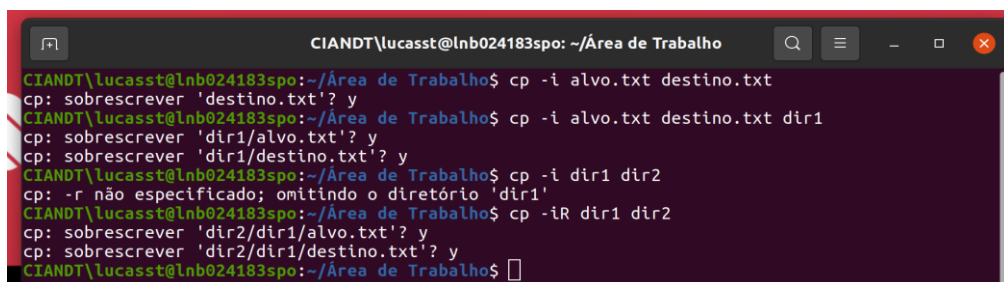
```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls
alvo.txt 'Linux - Ubuntu'
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ cp alvo.txt destino.txt
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```

Diretórios também podem ser copiados desde que usemos a tag de recursividade “-R”, todo o **conteúdo do diretório é copiado junto**:



```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir dir2
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ cp -vR dir1 dir2
'dir1' -> 'dir2/dir1'
'dir1/alvo.txt' -> 'dir2/dir1/alvo.txt'
'dir1/destino.txt' -> 'dir2/dir1/destino.txt'
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```

Se quiser ter certeza de que não vai sobrescrever nada importante, então usa a tag “-i” para confirmação:



```
CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ cp -i alvo.txt destino.txt
cp: sobrescrever 'destino.txt'? y
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ cp -i alvo.txt destino.txt dir1
cp: sobrescrever 'dir1/alvo.txt'? y
cp: sobrescrever 'dir1/destino.txt'? y
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ cp -i dir1 dir2
cp: -r não especificado; omitindo o diretório 'dir1'
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ cp -iR dir1 dir2
cp: sobrescrever 'dir2/dir1/alvo.txt'? y
cp: sobrescrever 'dir2/dir1/destino.txt'? y
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```



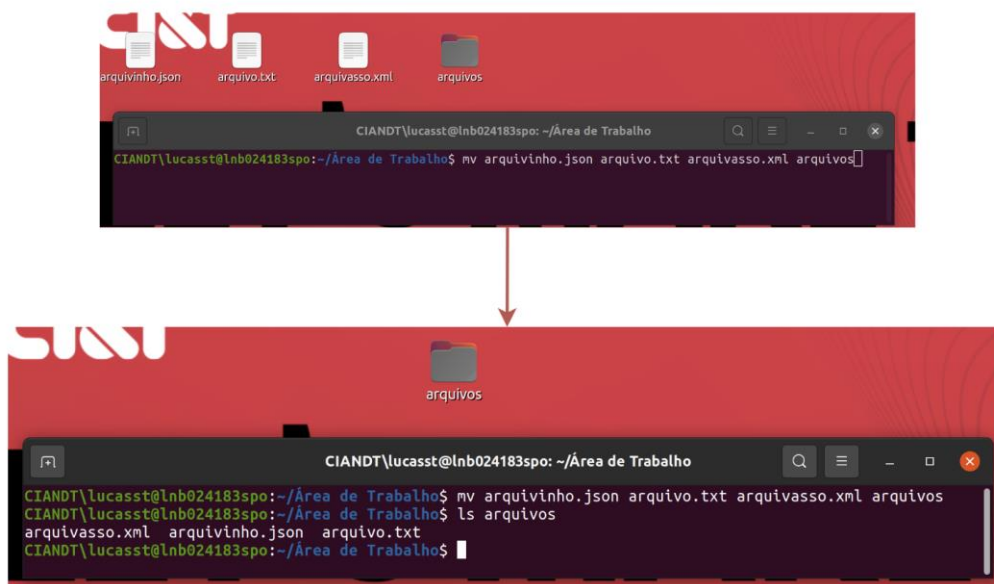
Movendo e renomeando arquivos – Linux

Com o comando “**mv xxx yyy**” você pode renomear um arquivo, sem cópia, é literalmente **renomear um arquivo específico**:



The image shows a terminal window with the prompt `CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho`. The first command entered is `mv nomes .nomesCompostos`. The second command is `ls -a`, which outputs the following: `.. 'Linux - Ubuntu' .nomesCompostos`. A red arrow points from the first terminal window to the second, indicating the sequence of actions.

O comando “**mv xxxx yyy**” também serve para mover fisicamente arquivos de um lugar para outro:



The image shows a desktop environment with a file manager window displaying four files: `arquivinho.json`, `arquivo.txt`, `arquivasso.xml`, and `arquivos`. Below the file manager is a terminal window with the prompt `CIANDT\lucasst@lnb024183spo: ~/Área de Trabalho`. The command entered is `mv arquivinho.json arquivo.txt arquivasso.xml arquivos`. A red arrow points from the file manager to the terminal, indicating the command used to move the files.

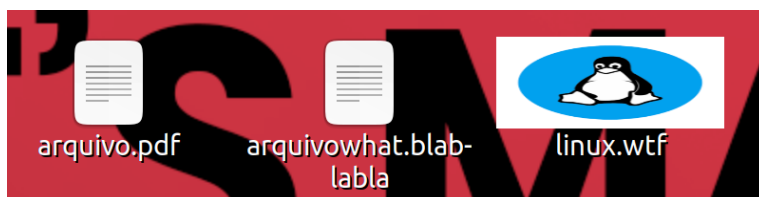
Esses

comandos podem ser usados com o mesmo propósito para diretórios.



Extensão de arquivos – Linux

No linux **não existe o conceito de extensão de arquivo**. A extensão do arquivo não afeta ele em nada, eu poderia mudar um arquivo “**.pdf**” para “**.blablabla**” e daria na mesma, ele ainda funcionaria no editor padrão do linux:



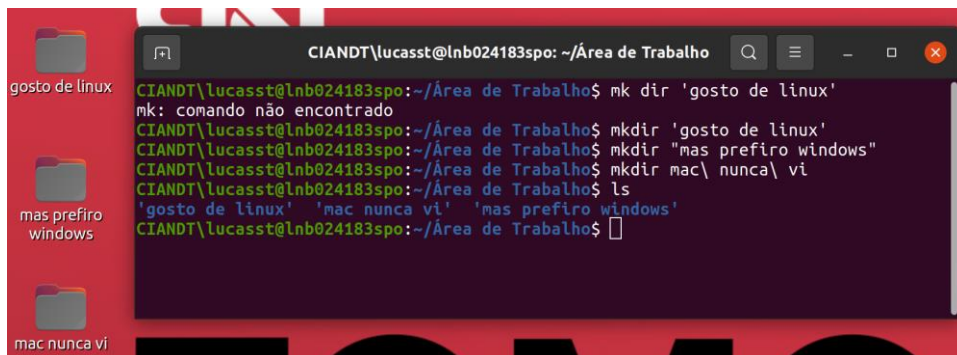
Então a ideia é que quando tiver um **arquivo que não tem uma extensão específica** (já que ela basicamente não significa nada) dá pra usar o comando **"file xxx"** e ele vai trazer informações a respeito do tipo de um arquivo:

```
arquivo.pdf arquivo.mat arquivo.txt linux - ubuntu linux.wtf
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ file linux.wtf
linux.wtf: PNG image data, 225 x 225, 8-bit colormap, non-interlaced
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```



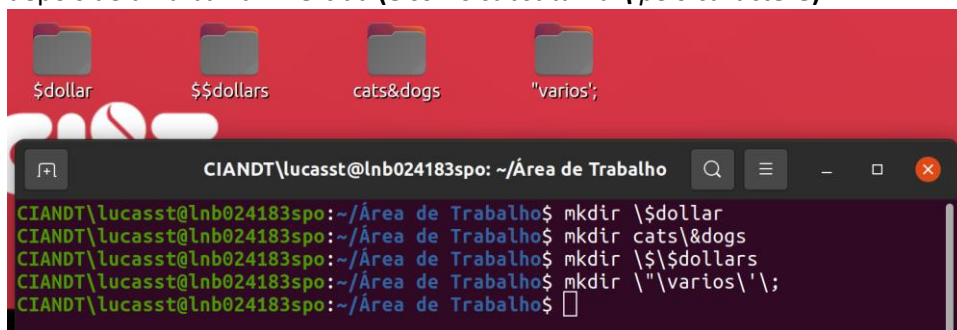
Espaço e caracteres especiais no nome – Linux

Se quiser criar um arquivo ou diretório com espaços em branco no nome, pode usar as **aspas simples** ou **aspas duplas** ao definir o nome, também pode usar a **barra invertida**:



```
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mk dir 'gosto de linux'
mk: comando não encontrado
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir 'gosto de linux'
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir "mas prefiro windows"
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir mac\ nunca\ vi
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ ls
'gosto de linux' 'mac nunca vi' 'mas prefiro windows'
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```

Caracteres especiais como: **> < & \$ | ; " ' ** podem ser incluídos como nome se forem passados depois de uma **barra invertida** (é como substituir a **** pelo **caractere**):



```
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir \$dollar
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir cats\&dogs
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir \$\$dollars
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$ mkdir "\"varios\"";
CIANDT\lucasst@lnb024183spo:~/Área de Trabalho$
```