



Terminal no Windows

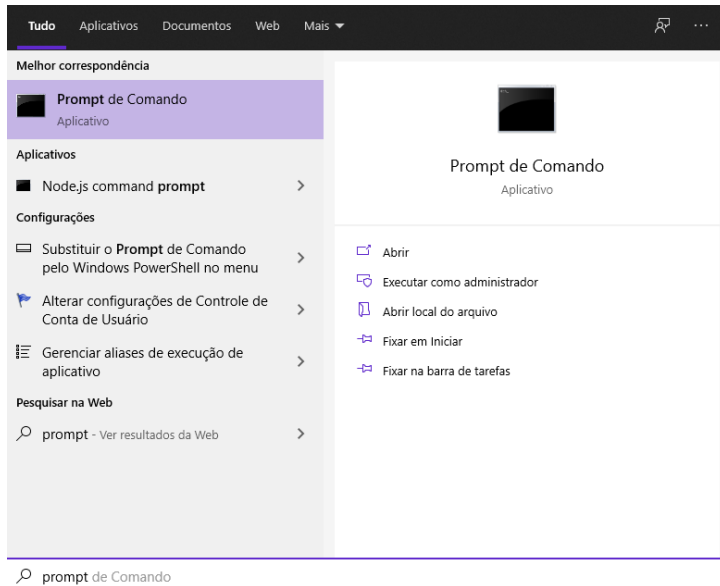
O Terminal no Windows é uma ferramenta imprescindível na vida dos profissionais da TI, do programador ao profissional de redes, pois é uma ferramenta capaz de fazer instalações, configurações, checar dados e navegar pelo sistema. O terminal não é uma novidade para ninguém no mundo da programação, ele sempre esteve ali e sempre estará, no entanto, não são todos que gostam ou sabem usá-lo. Caso esse seja seu caso, vamos te ajudar nesse artigo.

CMD vs PowerShell

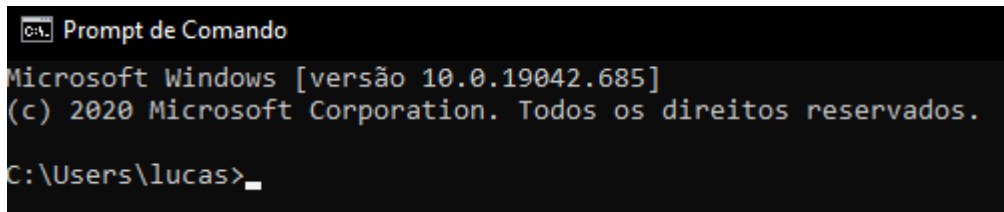
Atualmente, encontramos no Windows dois terminais diferentes. O **CMD** (ou Prompt de Comando) é o mais clássico, com fundo preto. Já o **Windows PowerShell** é mais recente e tem o fundo azul. Em resumo, o PowerShell é uma evolução do CMD, fazendo tudo o que o CMD faz e um pouco mais. Como por exemplo, usar alguns comandos próprios de sistemas baseados em Unix, ou seja, Linux e MacOS.

Conhecendo o CMD

Para abrir o CMD digite “CMD” ou “Prompt” no menu Iniciar:



Quando aberto dessa forma ele apontará diretamente para a pasta do seu usuário. Em outras palavras, no início o CMD vai apontar para o seguinte caminho: **C:**, que é o HD principal da sua máquina, onde o Windows está instalado; e **Users**, que é a pasta de usuários e seu nome de usuário.



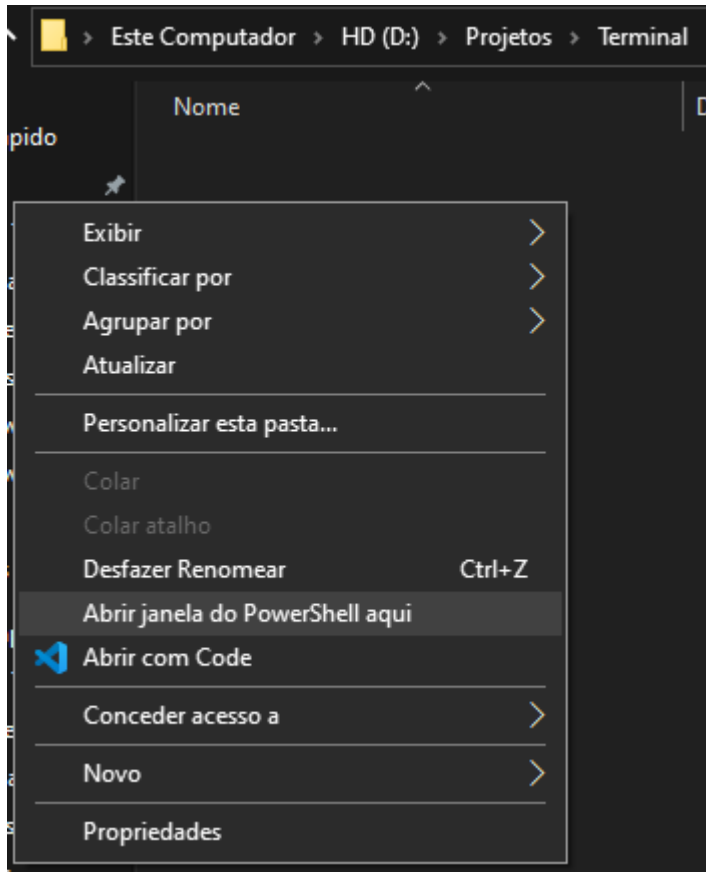
Para demonstrar melhor, vou usar o comando “**dir**“, que serve para mostrar as pastas e arquivos que estão na pasta apontada pelo terminal no Windows. Para utilizar o comando basta digitar e apertar Enter.

```
C:\Users\lucas>dir
O volume na unidade C é SSD
O Número de Série do Volume é ██████████

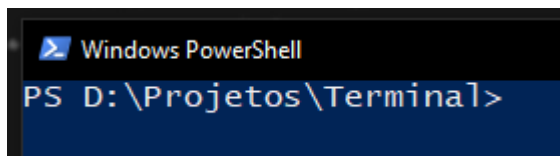
Pasta de C:\Users\lucas

06/01/2021  10:17    <DIR>      .
06/01/2021  10:17    <DIR>      ..
29/12/2020  11:34    <DIR>      .android
14/12/2020  14:07    <DIR>      .config
16/11/2020  11:36          188 .gitconfig
05/01/2021  10:42    <DIR>      .gradle
16/12/2020  11:44    <DIR>      .pylint.d
15/10/2020  16:27    <DIR>      .vscode
05/10/2020  19:52    <DIR>      3D Objects
11/10/2020  00:06    <DIR>      ansel
05/10/2020  19:52    <DIR>      Contacts
04/01/2021  18:34    <DIR>      Desktop
06/01/2021  10:17    <DIR>      Documents
06/01/2021  09:43    <DIR>      Downloads
23/10/2020  19:04          76.533 draws.bak
25/11/2020  14:54        107.520 draws.dwg
06/10/2020  11:19    <DIR>      Favorites
05/10/2020  19:52    <DIR>      Links
05/10/2020  19:52    <DIR>      Music
28/12/2020  18:37    <DIR>      Pictures
25/11/2020  14:48          685 plot.log
09/12/2020  21:25    <DIR>      Saved Games
05/10/2020  19:53    <DIR>      Searches
06/01/2021  07:01    <DIR>      Videos
               4 arquivo(s)          184.926 bytes
              20 pasta(s) 255.312.805.888 bytes disponíveis
```

Há ainda outra forma de abrir o terminal, mas dessa vez escolhendo a pasta onde ele irá apontar. Para isso, abra a pasta escolhida para iniciar o terminal no Windows, segure o SHIFT e clique com o botão DIREITO do mouse em alguma parte em branco da pasta. Atenção! Não clique em cima de nenhum arquivo ou pasta dentro daquela pasta. O resultado é o seguinte:



Dessa forma, você abrirá o PowerShell, mas como ele funciona da mesma forma não tem problema nenhum. Como resultado temos o terminal aberto já apontando para a pasta em questão para a qual navegamos.

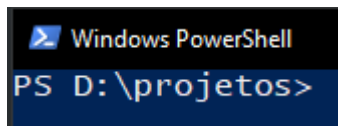


Navegando pelo sistema com o terminal

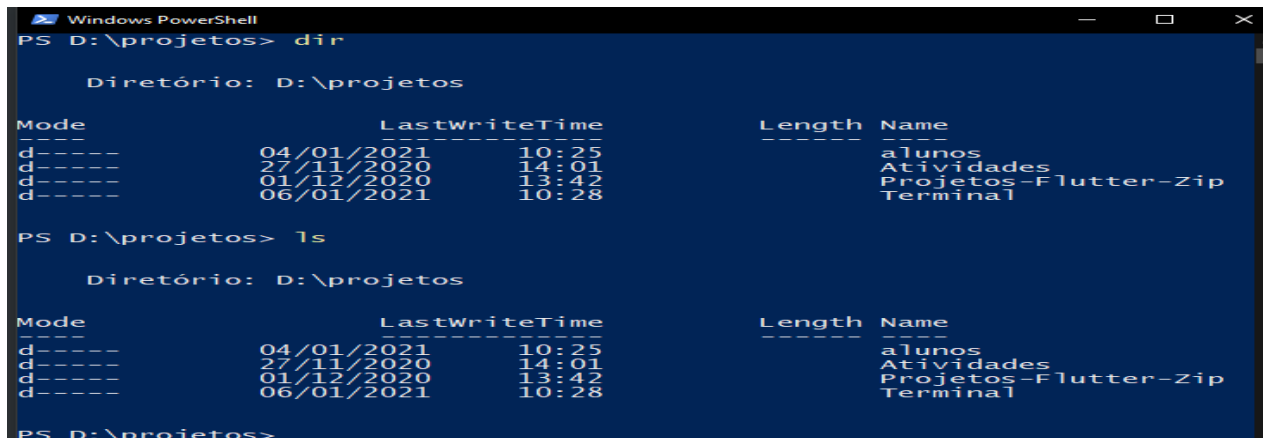
Vamos usar dois comandos para navegar pelas pastas do sistema: os comandos “cd” e “dir”. O “cd” é como um duplo clique no mouse, ou seja, ele abre a pasta, além de servir para voltar para trás.

Como ambos os terminais funcionam de forma semelhante, a partir daqui usarei apenas o PowerShell.

Com o terminal aberto na pasta selecionada, podemos usar o comando “**dir**” para descobrir o que há nela e poder avançar ou não. Para exemplificar melhor, começarei a partir da minha pasta “Projeto”, que fica no meu HD secundário.

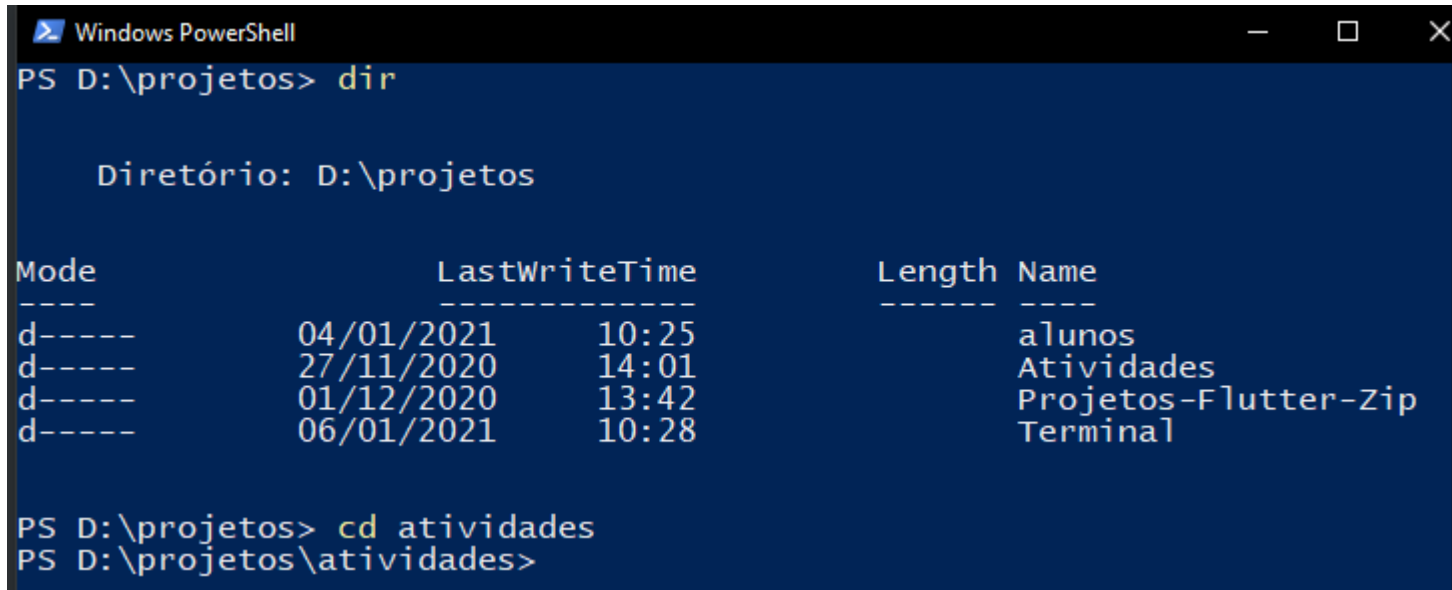


Como estamos no PowerShell, os comandos do Linux também funcionam. O comando equivalente ao “dir” no Linux é o “ls”, então também podemos usá-lo:



cd + nome da pasta

A partir daqui podemos usar o comando “cd + nome da pasta” para avançar para dentro de uma pasta. Nesse caso, você deve escrever o nome correto da pasta e ela deve estar dentro da pasta atual onde você está, por exemplo:



```
Windows PowerShell
PS D:\projetos> dir

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----         04/01/2021         10:25      alunos
d-----         27/11/2020         14:01      Atividades
d-----         01/12/2020         13:42      Projetos-Flutter-Zip
d-----         06/01/2021         10:28      Terminal

PS D:\projetos> cd atividades
PS D:\projetos\atividades>
```

Dentro da nova pasta podemos utilizar o comando “dir” novamente para verificar o que há nela.

```
Windows PowerShell
PS D:\projetos> dir

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----          04/01/2021         10:25      alunos
d-----          27/11/2020         14:01      Atividades
d-----          01/12/2020         13:42      Projetos-Flutter-Zip
d-----          06/01/2021         10:28      Terminal

PS D:\projetos> cd atividades
PS D:\projetos\atividades> dir

Diretório: D:\projetos\atividades

Mode                LastWriteTime         Length Name
----                -
-a----          19/11/2020         13:03      140761 Exercicios_de_Progra
-a----          27/11/2020         14:02      16144  mao_Resolucao.pdf
                                logica.py

PS D:\projetos\atividades> _
```

cd ..

Podemos usar o comando “cd ..” para retornar uma pasta atrás da atual. Esse comando é sempre o mesmo, você pode usá-lo sempre que quiser ir para a pasta anterior.

```
Windows PowerShell
PS D:\projetos> dir

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----          04/01/2021         10:25      alunos
d-----          27/11/2020         14:01      Atividades
d-----          01/12/2020         13:42      Projetos-Flutter-Zip
d-----          06/01/2021         10:28      Terminal

PS D:\projetos> cd atividades
PS D:\projetos\atividades> dir

Diretório: D:\projetos\atividades

Mode                LastWriteTime         Length Name
----                -
-a----          19/11/2020         13:03      140761 Exercicios_de_Progra
-a----          27/11/2020         14:02      16144  mao_Resolucao.pdf
                                logica.py

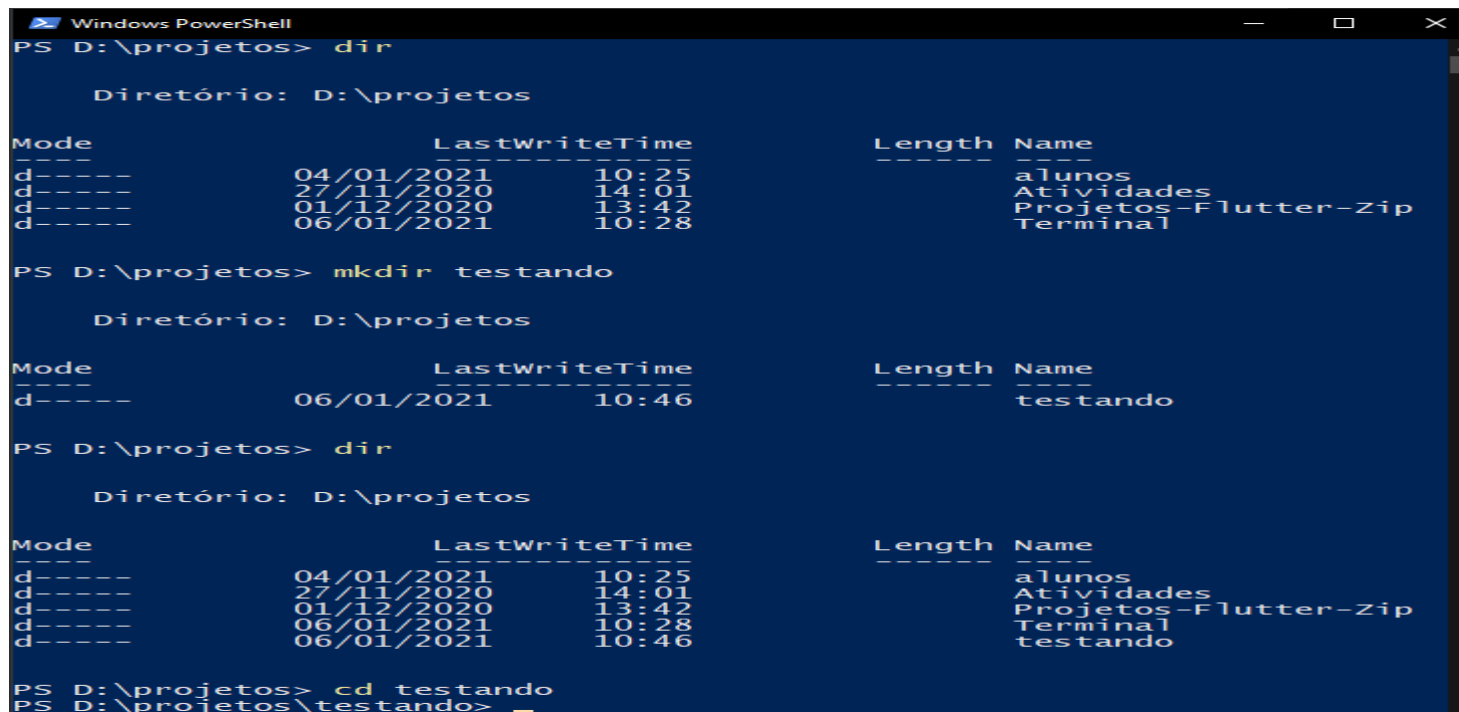
PS D:\projetos\atividades> cd ..
PS D:\projetos>
```

clear

Outro comando super interessante é o “cls” ou “clear”. Ele limpa todos os comandos listados, melhorando a visualização no terminal. Lembrando que ele limpa apenas os comandos, nenhum efeito de comando anterior é perdido.

mkdir

Você também pode criar uma nova pasta a partir do terminal com o comando “mkdir”. No exemplo a seguir, usei o “dir” para mostrar as pastas contidas na pasta “Projetos”, criei uma nova pasta, mostrei novamente as pastas com o “dir”, para mostrar que foi adicionada uma nova pasta lá, e entrei na pasta recém criada com o comando “cd”.



```
Windows PowerShell
PS D:\projetos> dir

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----          04/01/2021         10:25      alunos
d-----          27/11/2020         14:01      Atividades
d-----          01/12/2020         13:42      Projetos-Flutter-Zip
d-----          06/01/2021         10:28      Terminal

PS D:\projetos> mkdir testando

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----          06/01/2021         10:46      testando

PS D:\projetos> dir

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----          04/01/2021         10:25      alunos
d-----          27/11/2020         14:01      Atividades
d-----          01/12/2020         13:42      Projetos-Flutter-Zip
d-----          06/01/2021         10:28      Terminal
d-----          06/01/2021         10:46      testando

PS D:\projetos> cd testando
PS D:\projetos\testando> _
```


del + nome da pasta/arquivo

Você pode também apagar arquivos e pastas com o comando “del + nome da pasta/arquivo”. Da seguinte forma:

```
Windows PowerShell
PS D:\projetos> dir

Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----         04/01/2021         10:25      alunos
d-----         27/11/2020         14:01      Atividades
d-----         01/12/2020         13:42      Projetos-Flutter-Zip
d-----         06/01/2021         10:28      Terminal
d-----         06/01/2021         10:50      testando

PS D:\projetos> del testando
PS D:\projetos> dir

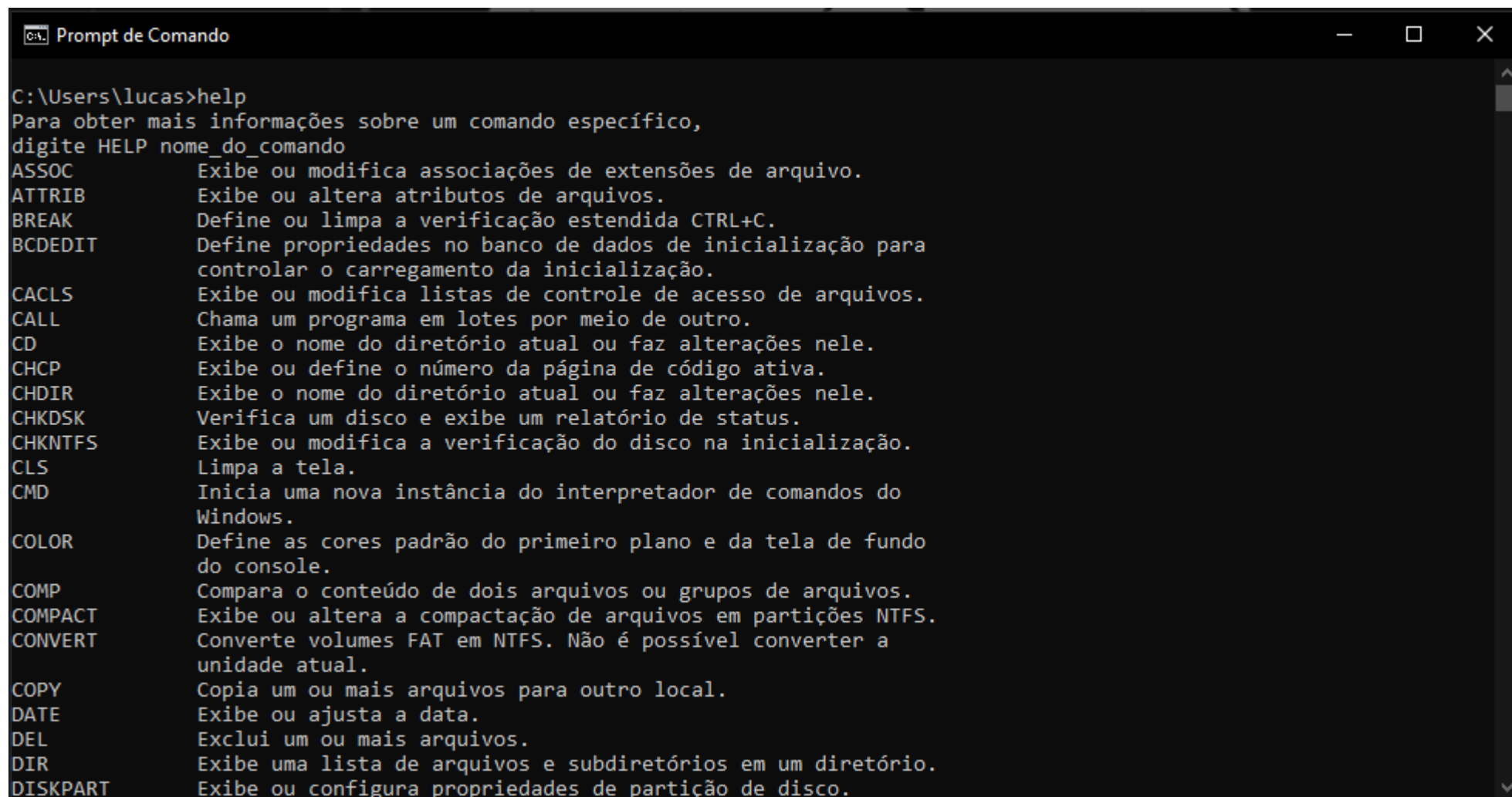
Diretório: D:\projetos

Mode                LastWriteTime         Length Name
----                -
d-----         04/01/2021         10:25      alunos
d-----         27/11/2020         14:01      Atividades
d-----         01/12/2020         13:42      Projetos-Flutter-Zip
d-----         06/01/2021         10:28      Terminal

PS D:\projetos>
```

Help

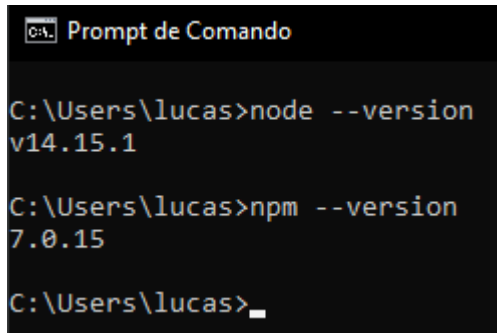
Um outro comando super válido de se utilizar é o ‘help’, pois ele lista todos os comandos que você pode utilizar. É muito bom para quem não quer ir pesquisar em outro local sobre os demais comandos do terminal. Basta utilizar o comando ‘help’ dentro dele mesmo e pronto!

A screenshot of a Windows Command Prompt window titled "Prompt de Comando". The window shows the output of the 'help' command, which lists various Windows command-line utilities and their functions. The text is displayed in a monospaced font on a dark background. The list includes commands like ASSOC, ATTRIB, BREAK, BCDEDIT, CACLS, CALL, CD, CHCP, CHDIR, CHKDSK, CHKNTFS, CLS, CMD, COLOR, COMP, COMPACT, CONVERT, COPY, DATE, DEL, DIR, and DISKPART, each followed by a brief description of its purpose.

```
C:\Users\lucas>help
Para obter mais informações sobre um comando específico,
digite HELP nome_do_comando
ASSOC           Exibe ou modifica associações de extensões de arquivo.
ATTRIB          Exibe ou altera atributos de arquivos.
BREAK           Define ou limpa a verificação estendida CTRL+C.
BCDEDIT         Define propriedades no banco de dados de inicialização para
                controlar o carregamento da inicialização.
CACLS           Exibe ou modifica listas de controle de acesso de arquivos.
CALL            Chama um programa em lotes por meio de outro.
CD              Exibe o nome do diretório atual ou faz alterações nele.
CHCP            Exibe ou define o número da página de código ativa.
CHDIR           Exibe o nome do diretório atual ou faz alterações nele.
CHKDSK          Verifica um disco e exibe um relatório de status.
CHKNTFS         Exibe ou modifica a verificação do disco na inicialização.
CLS             Limpa a tela.
CMD             Inicia uma nova instância do interpretador de comandos do
                Windows.
COLOR           Define as cores padrão do primeiro plano e da tela de fundo
                do console.
COMP            Compara o conteúdo de dois arquivos ou grupos de arquivos.
COMPACT         Exibe ou altera a compactação de arquivos em partições NTFS.
CONVERT         Converte volumes FAT em NTFS. Não é possível converter a
                unidade atual.
COPY            Copia um ou mais arquivos para outro local.
DATE            Exibe ou ajusta a data.
DEL            Exclui um ou mais arquivos.
DIR             Exibe uma lista de arquivos e subdiretórios em um diretório.
DISKPART        Exibe ou configura propriedades de partição de disco.
```

Outros usos

Várias tecnologias quando instaladas na sua máquina têm o potencial de serem utilizadas pelo terminal. Um exemplo disso é o [NodeJS](#), uma vez instalada na sua máquina você pode utilizar comandos com o ‘npm’ ou ‘node’, que indicam que você estará utilizando aquele software.



```
C:\Users\lucas>node --version
v14.15.1

C:\Users\lucas>npm --version
7.0.15

C:\Users\lucas>
```

Lembrando que no caso do Node é instalado um terminal próprio do Node que recebe comandos JavaScript e esse terminal do Node NÃO funciona como o CMD e o PowerShell do Windows, então não abram ele e tentem utilizar pois isso irá gerar erros nos comandos.

Outro caso é o [Flutter](#) e o [Dart](#), que também se utilizam do terminal após instalados na máquina, sendo capazes de criar projetos, se atualizar dentre outras coisas.

Esses outro comandos variam de tecnologia para tecnologia, de versão para versão, então não irei dar tantos exemplos aqui, mas sempre que se instalar uma nova tecnologia é bom lembrar de verificar e aprender quais comandos aquela tecnologia possui no seu terminal.

Esse pequeno guia é apenas um compilado de pequenas dicas para pessoas que tiveram pouco ou nenhum contato com o terminal consigam lidar de forma mais efetiva com essa ferramenta tão importante. E que também tenham menos medo dele! Espero ter ajudado. Bons estudos e até a próxima!

Terminal no MacOS e Linux

O Terminal no MacOS e Linux é uma ferramenta indispensável para os profissionais da TI, desde o programador até o profissional de redes.

Afinal, é uma ferramenta capaz de fazer instalações, configurações, checar dados e navegar pelo sistema.

Mesmo não sendo muito conhecido no mundo da programação, não são todos que gostam ou sabem usá-lo. Se esse é seu caso, vamos te ajudar nesse artigo!

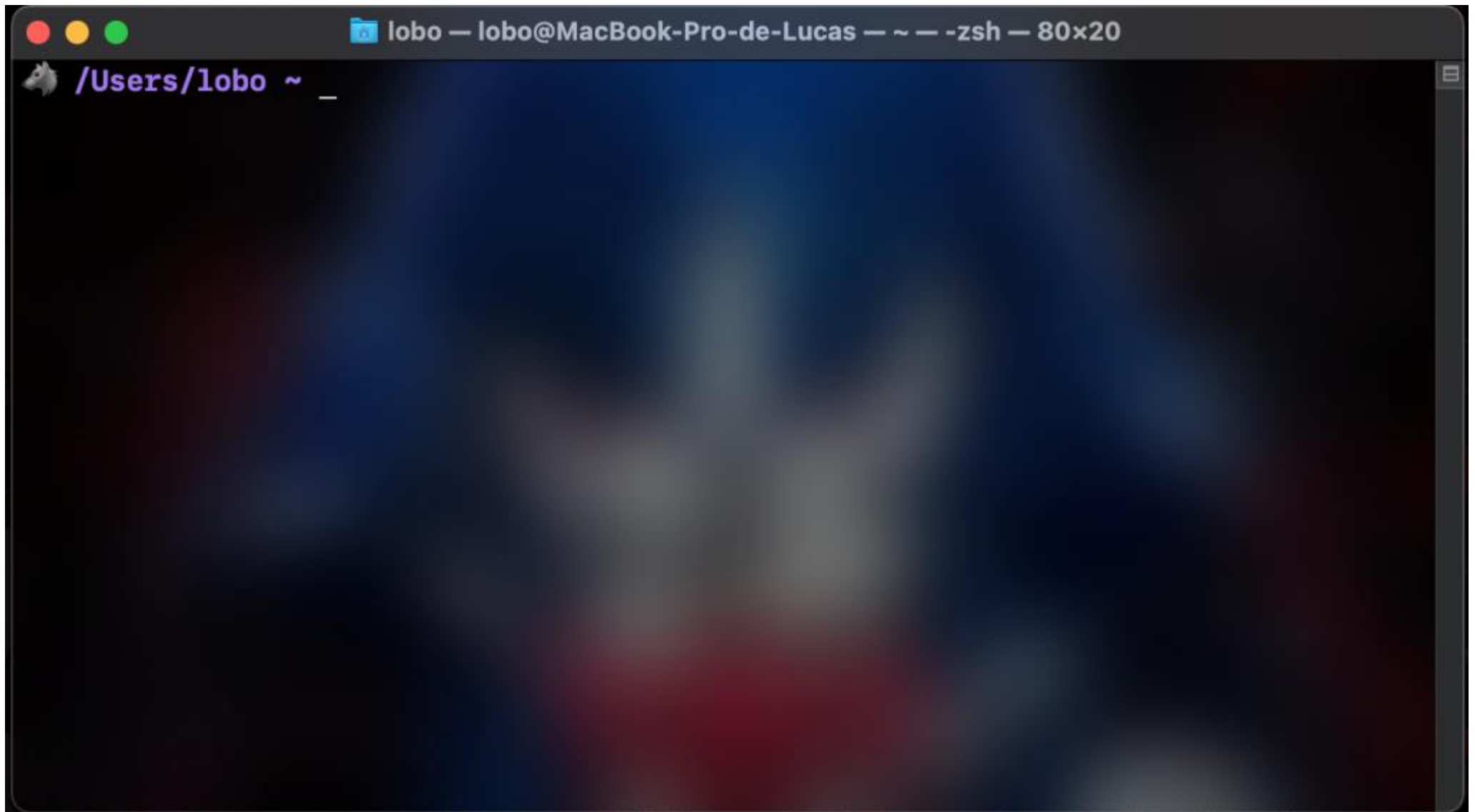
Conhecendo o Terminal

Para abrir o Terminal no MacOS e Linux procure por “Terminal” na sua lista de Programas/Aplicativos. No Mac, você pode usar o Spotlight com o atalho “COMMAND + ESPAÇO” e digitar “terminal” para abri-lo.

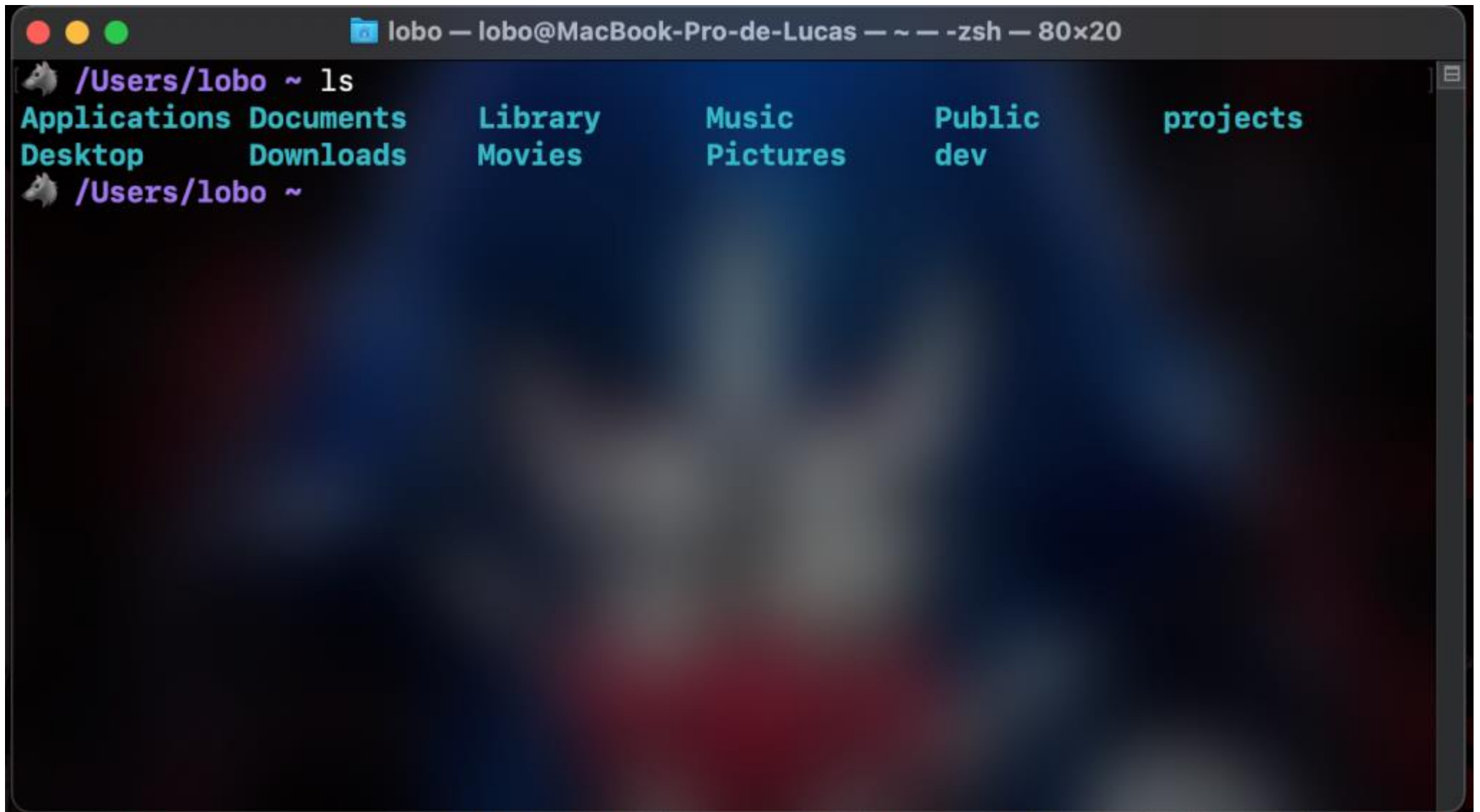


Abrindo dessa forma, ele apontará diretamente para a pasta do seu usuário, ou seja, para o caminho: C, que é o HD principal da sua máquina, onde o seu OS está instalado; e Users, que é a pasta de usuários e seu nome de usuário.

Pode ser que o que apareça na sua tela varie um pouco de acordo com seu sistema e a versão dele. Mas o padrão é o seguinte:



Para demonstrar melhor, vou usar o comando “ls”, que serve para mostrar as pastas e arquivos que estão na pasta apontada pelo terminal no momento. Para usar o comando basta digitar e apertar Enter.



```
lobo — lobo@MacBook-Pro-de-Lucas — ~ — -zsh — 80x20
/Users/lobo ~ ls
Applications  Documents    Library      Music        Public       projects
Desktop       Downloads    Movies       Pictures     dev
/Users/lobo ~
```

Você também pode escolher qual pasta será apontada ao abrir o Terminal. Basta abrir o explorador de Arquivos (no Mac é o Finder) e navegue até a pasta que você quer abra no terminal. Em seguida, clique com o botão direito na pasta e selecione a opção de abrir um novo Terminal na pasta.

No Mac você verá o seguinte:

Abrir em Nova Aba

Mover para o Lixo

Obter Informações

Renomear

Comprimir "github"

Duplicar

Criar Atalho

Visualização Rápida

Copiar

Compartilhar >

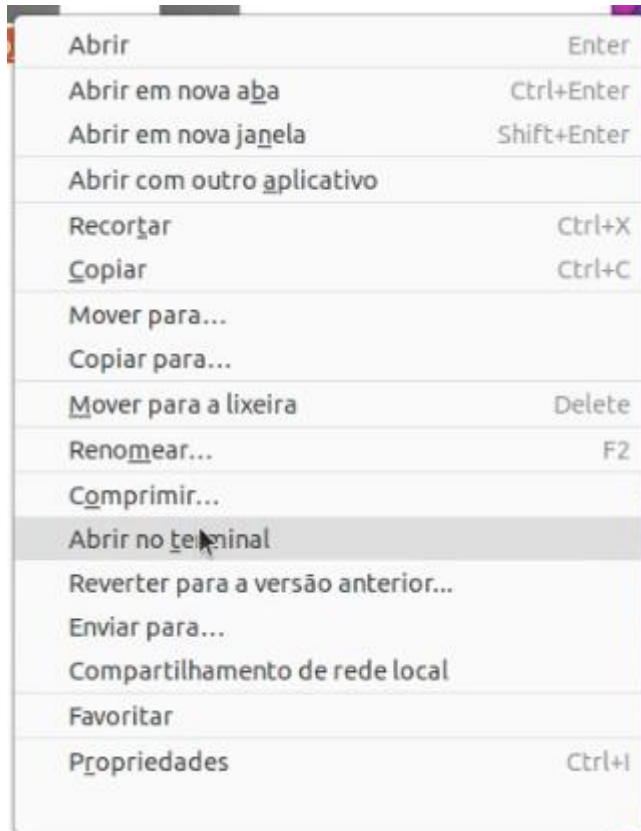


Etiquetas...

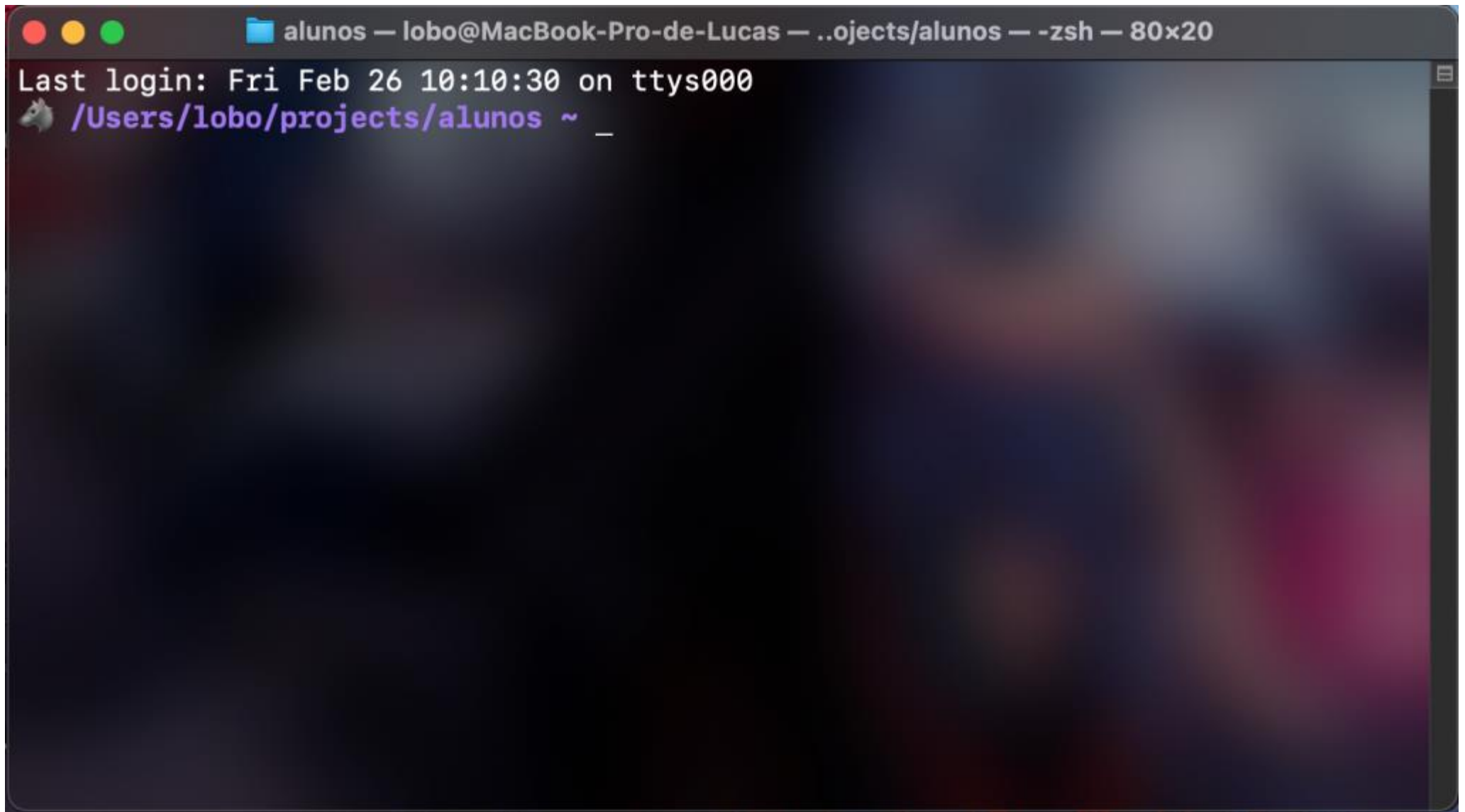
Ações Rápidas >

Confirmação de Ações de Destro

E no Ubuntu, assim:

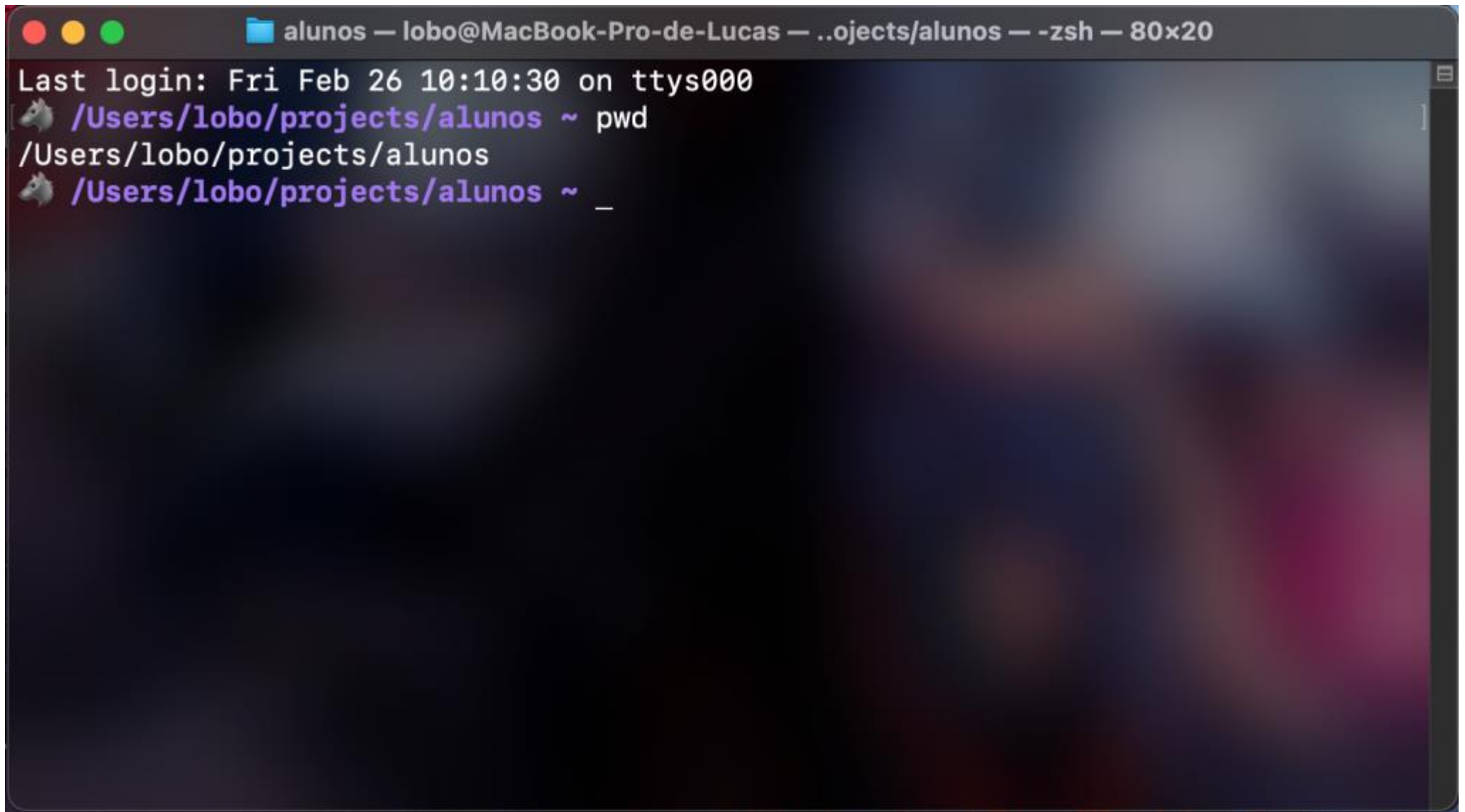


Ao clicar nessa opção, o Terminal abrirá diretamente naquela pasta.

A screenshot of a macOS terminal window. The title bar at the top shows three colored window control buttons (red, yellow, green) on the left, followed by a folder icon and the text "alunos — lobo@MacBook-Pro-de-Lucas — ../objects/alunos — -zsh — 80x20". The terminal content shows "Last login: Fri Feb 26 10:10:30 on ttys000" followed by a prompt consisting of a small grey wolf head icon, the path "/Users/lobo/projects/alunos" in purple, a tilde "~", and a cursor "_".

```
alunos — lobo@MacBook-Pro-de-Lucas — ../objects/alunos — -zsh — 80x20
Last login: Fri Feb 26 10:10:30 on ttys000
🐺 /Users/lobo/projects/alunos ~ _
```

Se o caminho para a pasta não aparecer no terminal, você pode testar se deu tudo certo com o comando “pwd”. Esse comando printa o caminho atual no terminal, ou seja, para onde ele está apontando.

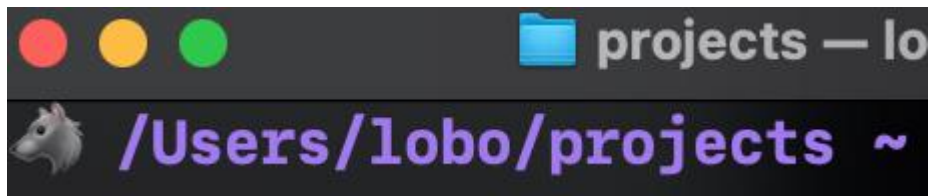
A screenshot of a macOS terminal window. The title bar at the top shows three colored window control buttons (red, yellow, green) on the left, followed by a folder icon and the text "alunos — lobo@MacBook-Pro-de-Lucas — ../objects/alunos — -zsh — 80x20". The terminal content shows a login message: "Last login: Fri Feb 26 10:10:30 on ttys000". Below this, a prompt character (a small grey horse head icon) is followed by the path "/Users/lobo/projects/alunos" in purple, a tilde "~", and the command "pwd". The output of the command is "/Users/lobo/projects/alunos". Another prompt character is followed by the same path in purple, a tilde, and an underscore "_".

```
alunos — lobo@MacBook-Pro-de-Lucas — ../objects/alunos — -zsh — 80x20
Last login: Fri Feb 26 10:10:30 on ttys000
🐾 /Users/lobo/projects/alunos ~ pwd
/Users/lobo/projects/alunos
🐾 /Users/lobo/projects/alunos ~ _
```

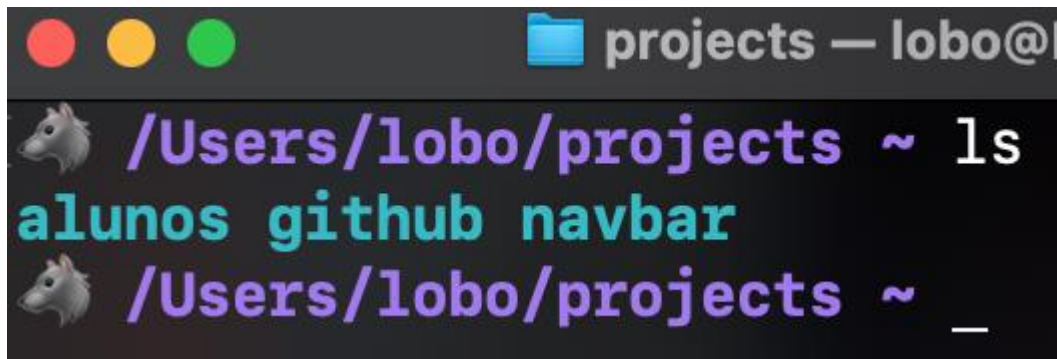
Navegando pelo sistema com o terminal

Vamos usar dois comandos para navegar pelas pastas do sistema: os comandos “cd” e “ls”. O “cd” funciona como um duplo clique no mouse, ou seja, ele abre a pasta e retorna para a pasta anterior.

O “ls”, quando o terminal está aberto na pasta selecionada, serve para descobrir o que há nela e avançar ou não. Para exemplificar melhor, começarei a partir da minha pasta “projects”, que fica no meu HD principal.



Primeiramente, vamos usar o comando “ls” para saber quais pastas estão dentro da pasta atual.



Você também pode usar algumas variações do comando ls, como por exemplo a flag “a” que mostra todos os arquivos, inclusive os escondidos.

```
projects — lobo@MacBook-Pro-de-Lucas — ~/projects —
/Users/lobo/projects ~ ls -a
.      ..      .DS_Store alunos  github  navbar
/Users/lobo/projects ~
```

Outra flag é “l”, que mostra a versão longa do “ls” normal, ou seja, traz mais informações sobre os arquivos.

```
projects — lobo@MacBook-Pro-de-Lucas — ~/proj
/Users/lobo/projects ~ ls -l
total 0
drwxr-xr-x  5 lobo  staff  160 28 Jan 17:09 alunos
drwxr-xr-x  4 lobo  staff  128 18 Feb 23:16 github
drwxr-xr-x 15 lobo  staff  480 16 Feb 17:37 navbar
/Users/lobo/projects ~ _
```

Você também pode juntar as flags e usá-las ao mesmo tempo:

```
projects — lobo@MacBook-Pro-de-Lucas — ~/projects —  
🐾 /Users/lobo/projects ~ ls -al  
total 24  
drwxr-xr-x@  6 lobo  staff   192 17 Feb 16:39 .  
drwxr-xr-x+ 34 lobo  staff 1088 26 Feb 10:53 ..  
-rw-r--r--@  1 lobo  staff 8196 26 Feb 10:38 .DS_Store  
drwxr-xr-x   5 lobo  staff  160 28 Jan 17:09 alunos  
drwxr-xr-x   4 lobo  staff  128 18 Feb 23:16 github  
drwxr-xr-x  15 lobo  staff  480 16 Feb 17:37 navbar  
🐾 /Users/lobo/projects ~
```

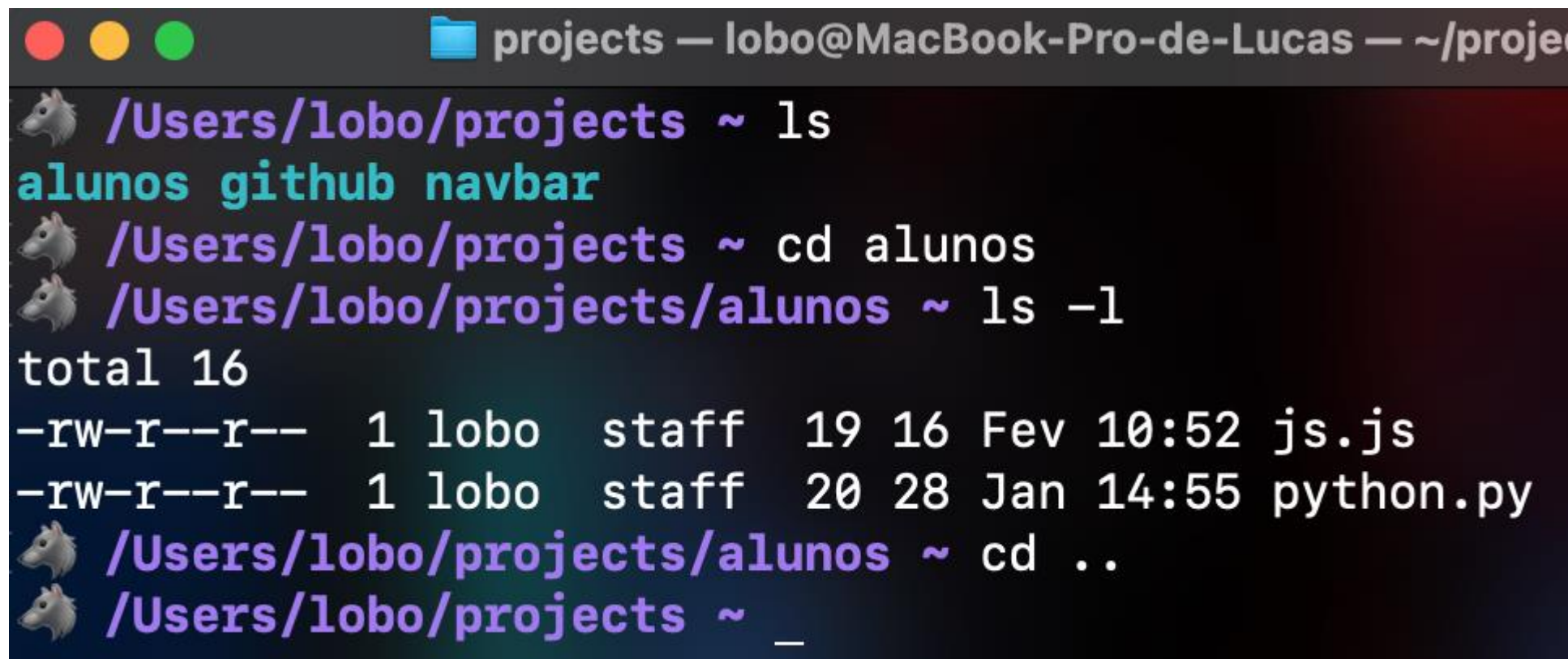
De agora em diante, podemos usar o comando “cd + nome da pasta” para avançar para dentro de uma pasta. Nesse caso, você deve escrever o nome correto da pasta e ela deve estar dentro da pasta atual onde você está, por exemplo:


```
alunos — lobo@MacBook-Pro-  
🐾 /Users/lobo/projects ~ ls  
alunos github navbar  
🐾 /Users/lobo/projects ~ cd alunos  
🐾 /Users/lobo/projects/alunos ~ _
```

Dentro da nova pasta podemos usar o comando “ls” para verificar seu conteúdo:

```
alunos — lobo@MacBook-Pro-de-Lucas — ../objects/alu  
🐾 /Users/lobo/projects ~ ls  
alunos github navbar  
🐾 /Users/lobo/projects ~ cd alunos  
🐾 /Users/lobo/projects/alunos ~ ls -l  
total 16  
-rw-r--r--  1 lobo  staff  19 16  Fev 10:52  js.js  
-rw-r--r--  1 lobo  staff  20 28  Jan 14:55  python.py  
🐾 /Users/lobo/projects/alunos ~ _
```

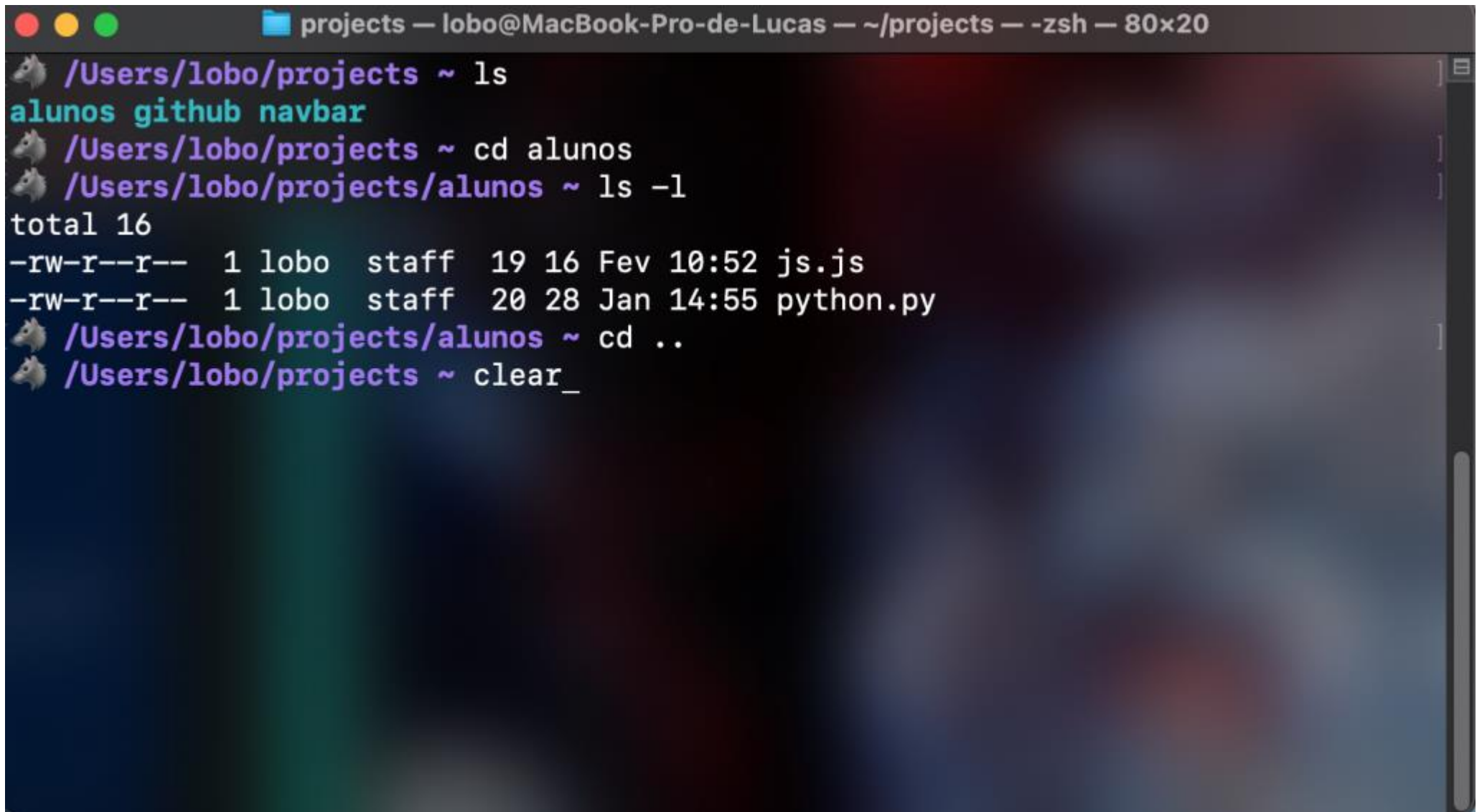
Podemos usar também o comando “cd ..” para retornar para a pasta anterior.

A screenshot of a macOS Terminal window. The title bar shows three colored window control buttons (red, yellow, green) on the left, a folder icon, and the text "projects — lobo@MacBook-Pro-de-Lucas — ~/projects". The terminal content shows a series of commands and their outputs. The prompt is a horse head emoji. The commands are: "ls", "cd alunos", "ls -l", "cd ..", and a final prompt. The output of "ls -l" shows a directory listing for "alunos" with two files: "js.js" and "python.py".

```
🐾 /Users/lobo/projects ~ ls
alunos github navbar
🐾 /Users/lobo/projects ~ cd alunos
🐾 /Users/lobo/projects/alunos ~ ls -l
total 16
-rw-r--r--  1 lobo  staff   19  16  Fev  10:52  js.js
-rw-r--r--  1 lobo  staff   20  28  Jan  14:55  python.py
🐾 /Users/lobo/projects/alunos ~ cd ..
🐾 /Users/lobo/projects ~ _
```

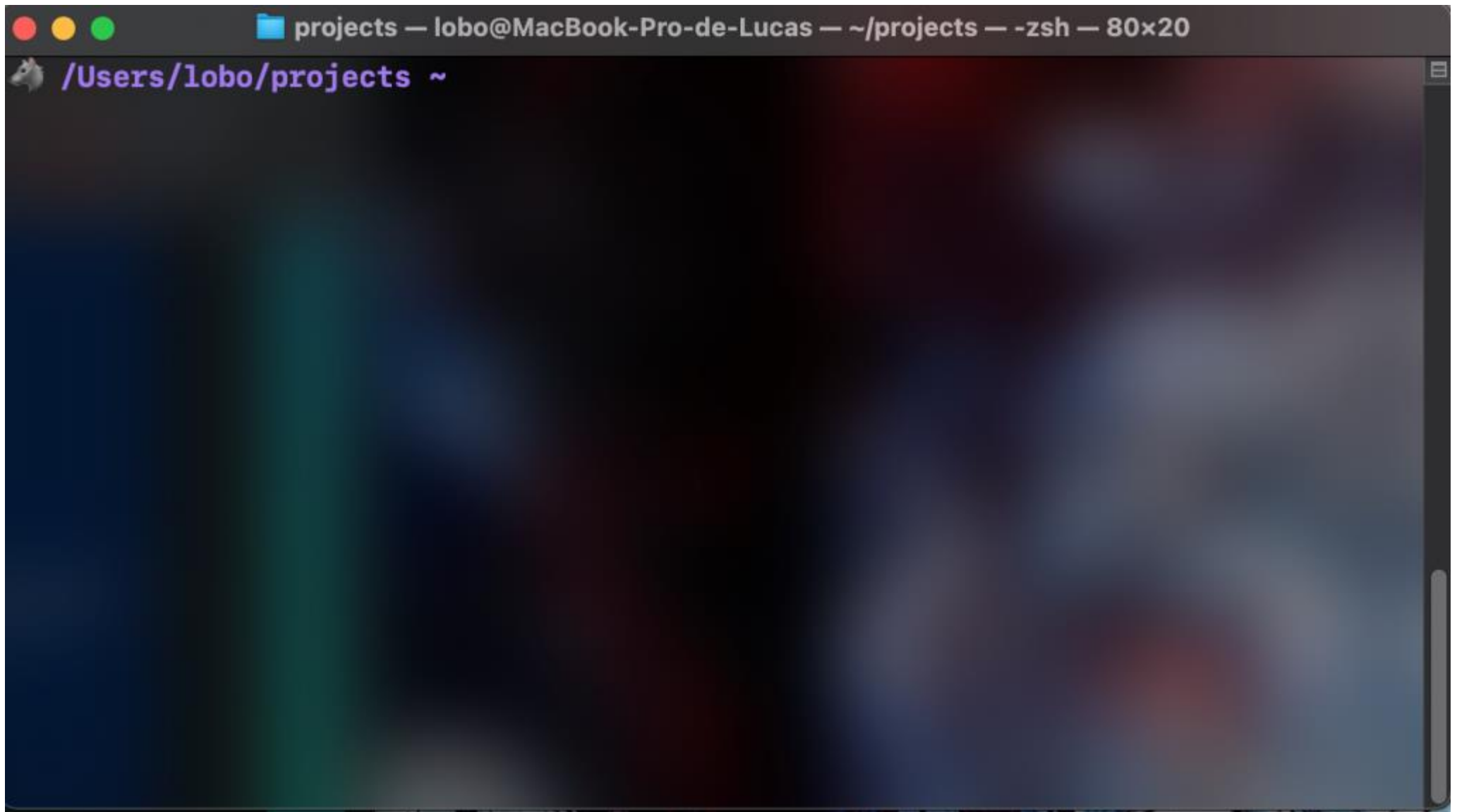
Limpendo o Terminal

Após usarmos vários comandos, um após o outro, o Terminal acaba ficando poluído.



```
projects — lobo@MacBook-Pro-de-Lucas — ~/projects — -zsh — 80x20
/Users/lobo/projects ~ ls
alunos github navbar
/Users/lobo/projects ~ cd alunos
/Users/lobo/projects/alunos ~ ls -l
total 16
-rw-r--r--  1 lobo  staff   19 16  Feb 10:52 js.js
-rw-r--r--  1 lobo  staff   20 28  Jan 14:55 python.py
/Users/lobo/projects/alunos ~ cd ..
/Users/lobo/projects ~ clear_
```

Por isso usamos o comando “clear” para limpar a tela.

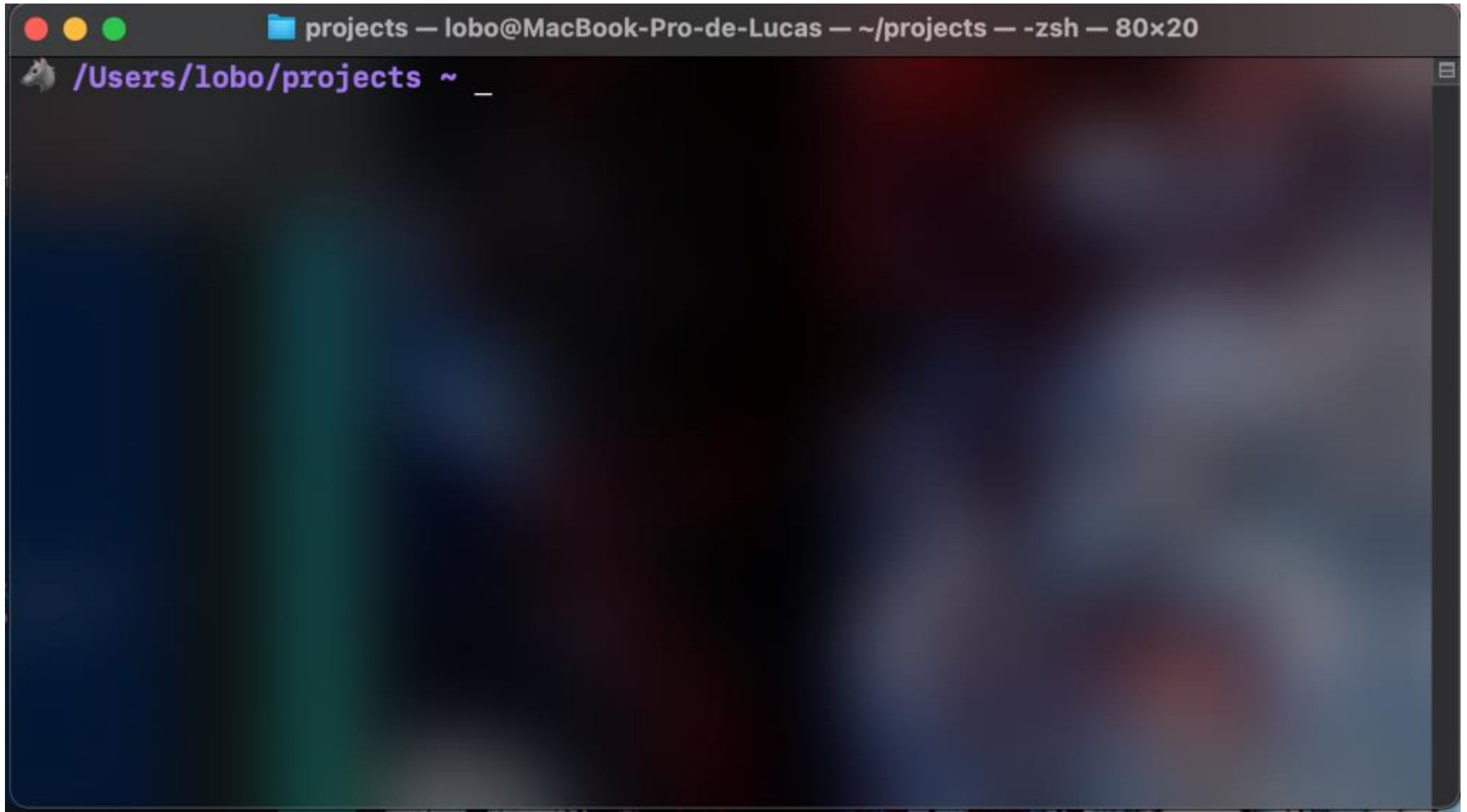


Um fato interessante é que no MacOS o comando “clear” não limpa de fato o terminal, mas apenas printa várias linhas vazias para que os comandos usados anteriormente subam na tela e não fiquem visíveis.

```
🐾 /Users/lobo/projects ~ ls
alunos github navbar
🐾 /Users/lobo/projects ~ cd alunos
🐾 /Users/lobo/projects/alunos ~ ls -l
total 16
-rw-r--r--  1 lobo  staff  19 16  Feb 10:52 js.js
-rw-r--r--  1 lobo  staff  20 28  Jan 14:55 python.py
🐾 /Users/lobo/projects/alunos ~ cd ..
🐾 /Users/lobo/projects ~ clear
```

```
🐾 /Users/lobo/projects ~ _
```

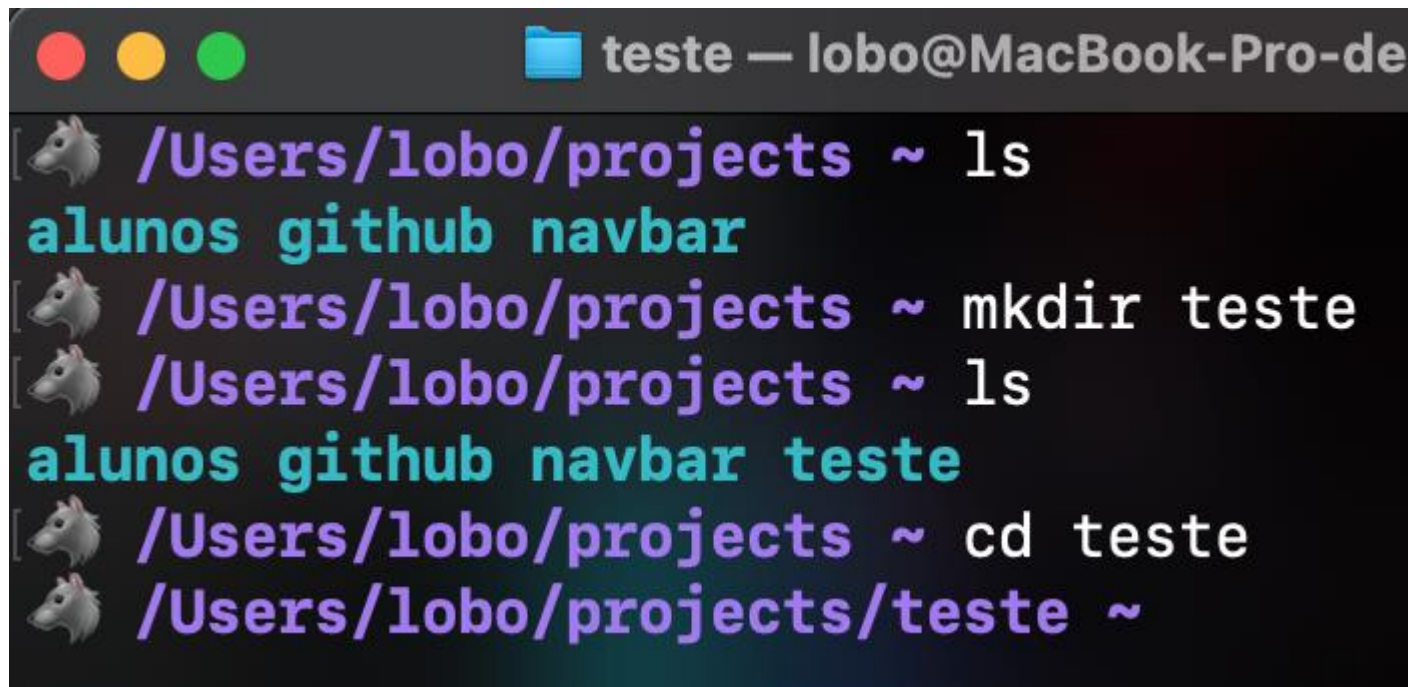
Para limpar o terminal de verdade, você deve usar o atalho “COMMAND + K”. Dessa forma o terminal será de fato zerado e renovado.



Criando pastas via terminal

Você também pode criar uma nova pasta a partir do terminal com o comando “mkdir”.

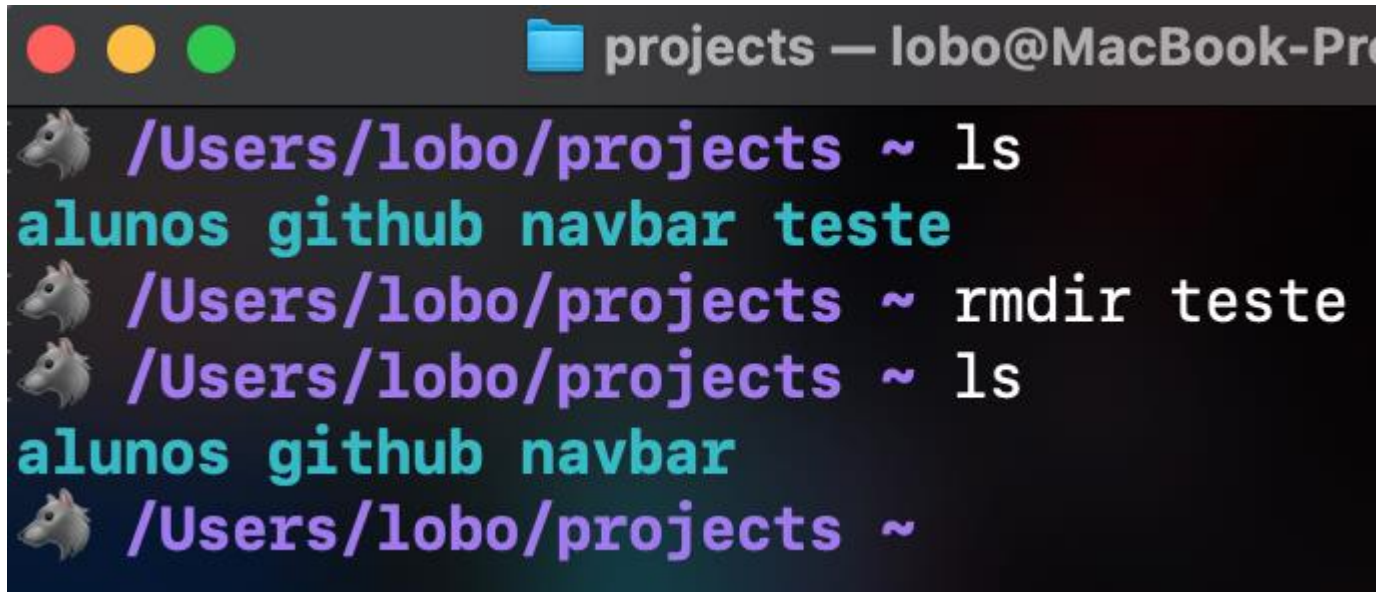
No exemplo abaixo eu usei o “ls” para mostrar as pastas contidas dentro na pasta “projects”; criei uma nova pasta; mostrei novamente as pastas com o “ls”, para mostrar que foi adicionada uma nova pasta lá; e entrei na pasta recém criada com o comando “cd”.

A screenshot of a macOS terminal window. The title bar at the top shows three colored window control buttons (red, yellow, green) on the left, followed by a blue folder icon and the text "teste — lobo@MacBook-Pro-de...". The terminal has a dark background with light-colored text. It shows a series of commands and their outputs, each preceded by a prompt character and a horse head emoji. The commands are: 1. `/Users/lobo/projects ~ ls` with output `alunos github navbar`; 2. `/Users/lobo/projects ~ mkdir teste`; 3. `/Users/lobo/projects ~ ls` with output `alunos github navbar teste`; 4. `/Users/lobo/projects ~ cd teste`; 5. `/Users/lobo/projects/teste ~` (no output shown).

```
[🐾 /Users/lobo/projects ~ ls
alunos github navbar
[🐾 /Users/lobo/projects ~ mkdir teste
[🐾 /Users/lobo/projects ~ ls
alunos github navbar teste
[🐾 /Users/lobo/projects ~ cd teste
[🐾 /Users/lobo/projects/teste ~
```

Deletando pastas e arquivos via terminal

Para deletar arquivos pelo terminal basta usar o comando “rm” para deletar arquivos, e “rmdir” para deletar pastas. Vou exemplificar deletando a pasta que criei no exemplo anterior.

A screenshot of a macOS terminal window. The title bar at the top shows three colored window control buttons (red, yellow, green) on the left, followed by a folder icon and the text 'projects — lobo@MacBook-Pro'. The terminal content shows a series of commands and their outputs. The prompt is a white horse head emoji. The commands are: 1. `/Users/lobo/projects ~ ls` followed by the output `alunos github navbar teste` on the next line. 2. `/Users/lobo/projects ~ rmdir teste`. 3. `/Users/lobo/projects ~ ls` followed by the output `alunos github navbar` on the next line. 4. `/Users/lobo/projects ~` followed by a blank line.

```
projects — lobo@MacBook-Pro  
🐾 /Users/lobo/projects ~ ls  
alunos github navbar teste  
🐾 /Users/lobo/projects ~ rmdir teste  
🐾 /Users/lobo/projects ~ ls  
alunos github navbar  
🐾 /Users/lobo/projects ~
```

Outros usos

Várias tecnologias instaladas na sua máquina podem ser usadas pelo terminal, como por exemplo o NodeJS, Flutter e Dart. Após instalar, você pode usar os comandos com o ‘npm’, ‘node’, ‘dart’ ou ‘flutter’ para indicar que você quer usar aquele software especificamente.


```
projects — lobo@MacBook-Pro-de-Lucas — ~/projects — -zsh — 80x20

/Users/lobo/projects ~ npm --version
6.14.10

/Users/lobo/projects ~ node --version
v14.15.4

/Users/lobo/projects ~ flutter --version
Flutter 1.22.6 • channel stable • https://github.com/flutter/flutter.git
Framework • revision 9b2d32b605 (5 weeks ago) • 2021-01-22 14:36:39 -0800
Engine • revision 2f0af37152
Tools • Dart 2.10.5

/Users/lobo/projects ~ dart --version
Dart SDK version: 2.10.5 (stable) (Tue Jan 19 13:05:37 2021 +0100) on "macos_x64"

/Users/lobo/projects ~ _
```

Esses outros comandos variam de tecnologia para tecnologia, de versão para versão, então não irei dar tantos exemplos aqui. Mas sempre que você instalar uma nova tecnologia, é bom verificar quais comandos podem ser usados no terminal.

Nesses dois sites existem alguns links com vários comandos e dicas que você pode usar no terminal:

<https://www.makeuseof.com/tag/mac-terminal-commands-cheat-sheet/> <https://www.guru99.com/linux-commands-cheat-sheet.html>

Esse pequeno guia é apenas um compilado de pequenas dicas para pessoas que tiveram pouco ou nenhum contato com o terminal no MacOS e Linux consigam lidar de forma mais efetiva com essa ferramenta tão importante. E que também tenham menos medo dele! Espero ter ajudado.