

# <COMANDOS INICIAIS>



## { Introdução

Um sistema operacional é um programa (ou *software*) que fornece interface entre o usuário e o *hardware*. O computador não pode executar nenhuma tarefa sem um sistema operacional, que controla e gerencia a operação pelo hardware do computador.

Os tópicos compartilhados abaixo nos ajudarão a entender a importância de nos familiarizarmos com comandos utilizados pelo sistema operacional Linux e sua base histórica.

Hoje, embora o Linux tenha uma participação de mercado mundial de 2,68% em *desktops*, mais de 90% de toda infraestrutura em nuvem e serviços de hospedagem é executada nesse sistema operacional. De acordo com a pesquisa [StackOverflow de 2021](#), o Linux também é um dos sistemas operacionais mais utilizados por desenvolvedores de sistemas. Por esse motivo, é crucial a familiarização com os comandos populares do Linux.

## { O que é CLI?

A Interface de Linha de Comando (ou CLI, que significa *Command-Line Interface* em inglês) é um programa que permite aos usuários digitar comandos de texto para instruir o computador a realizar tarefas específicas.

## { O que é GUI?

A Interface Gráfica do Usuário (ou GUI, que significa *Graphical User Interface* em inglês) foi desenvolvida dentro do sistema operacional logo que o mouse se tornou um dispositivo de entrada para operar o computador. A GUI fornece interface gráfica para o usuário trabalhar de forma mais interativa e visual, por meio de um ambiente mais amigável. O usuário pode trabalhar clicando em ícones e abrindo o arquivo sem precisar escrever nenhum comando.



## { O início de tudo

Na década de 1960, a CLI passou a ser usada intensamente, já que usuários tinham apenas um teclado como dispositivo de entrada e a tela do computador só exibia informações de texto.

Em sistemas operacionais como o MS-DOS, que usavam CLI como interface de usuário padrão, os usuários precisavam digitar um comando para realizar tarefas, uma vez que essa era a única maneira de se comunicar com o computador.

Depois de digitar um comando, o usuário recebia uma informação de texto ou ação específica executada pelo computador. Se digitasse o comando errado, corria o risco de excluir arquivos

ou fechar acidentalmente o programa antes de salvar seu trabalho. De acordo com usuários, essa era a principal desvantagem da CLI.

Após anos usando apenas um teclado e linhas de comandos para navegação, modificação e exclusão de pastas, usuários assistiram à invenção do mouse e o início do clique como nova forma de interagir com o computador.

Hoje, embora o uso da GUI tenha se tornado comum na área da computação, a maior parte dos sistemas operacionais ainda oferece uma combinação de CLI e GUI. Usuários de Mac podem digitar `cal` no Terminal ou clicar em um aplicativo de calendário para obter os mesmos resultados.

## { Unix

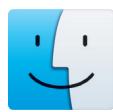
O Unix é um sistema operacional portátil multitarefas e multiusuário, que roda independente da arquitetura enquanto executa múltiplas tarefas de múltiplos usuários simultaneamente. A maioria dos servidores ou provedores *cloud* utilizam esse sistema ou o Linux, embora o Unix tenha sido pioneiro dentre os sistemas operacionais, já que serviu de base para vários sistemas subsequentes.



## { Sistemas operacionais baseados no UNIX



• Linux



• MacOs



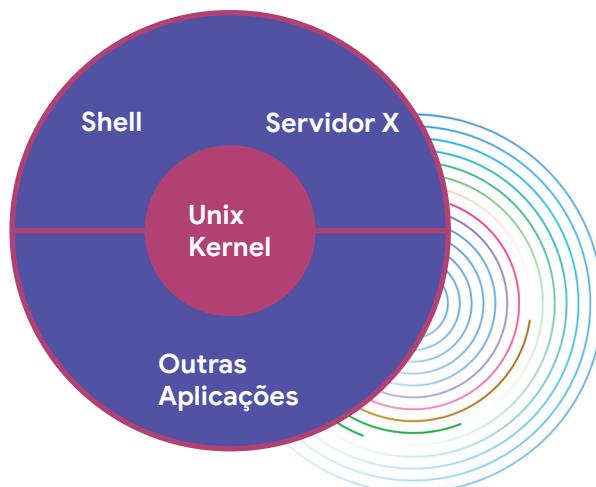
• Solaris



• BSD

## { Estrutura

O Unix possui uma organização própria, como todo sistema operacional. A imagem a seguir ilustra melhor sua estrutura.



• **Kernel:** Núcleo do sistema operacional, o Kernel é a parte interna que trabalha diretamente sobre o *hardware* e traduz comandos do usuário para instruções de máquina. O Kernel não interage com o usuário do sistema.

• **Shell:** Programa que atua como interface entre o Kernel e o usuário, o Shell funciona com linha de comando recebendo comandos digitados pelo usuário.

• **Servidor X:** Interface gráfica implementada nas versões mais recentes do Unix.

• **Outras aplicações:** Programas invocados pelo Shell para diversas tarefas.

Nessa divisão, também consta o sistema de arquivos.

## { Um pouco mais sobre o Shell

Se mergulharmos da CLI para a parte mais profunda de um sistema operacional, encontraremos a Shell.

A Shell é uma interface de usuário responsável por processar todos os comandos digitados na CLI. Ela lê e interpreta comandos instruindo o sistema operacional a executar tarefas conforme solicitado. Em outras palavras, a Shell gerencia a CLI e atua como intermediária conectando usuários ao sistema operacional.

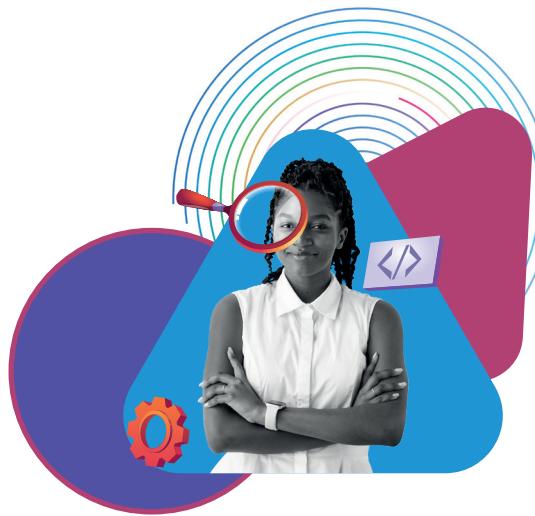
Na prática, a Shell processa múltiplas tarefas, como:

- Trabalhar com arquivos e diretórios
- Abrir e fechar um programa
- Gerenciar processos de computador
- Executar tarefas repetitivas

Dentre os muitos tipos de Shell, os mais populares são o do Windows (para PC) e o Bash (para Linux e MacOS).

## { O que é linha de comando?

A CLI como interface de texto permite navegação além do uso do mouse, por digitação de comandos (linhas de texto processadas como instruções para o computador) através da GUI. A CLI também é chamada de “*prompt de comando*” (no sistema operacional Windows), “terminal” (no Linux), “tela de comando”, Shell ou Bash.



## { Estrutura de comandos Linux

O **Flag** é um mecanismo lógico de controle de programa usualmente opcional que interrompe ou permite a execução de comandos. Grande parte dos comandos do Linux possui uma página de ajuda acionável pelo sinalizador “-h”.

```
$ ls
```

Um **argument** (ou parâmetro) se refere à criação de um comando que permite que ele seja executado corretamente. Na maioria dos casos, o **argument** é um caminho de arquivo.

```
$ ls -la
```

Você pode utilizar o **flag** por meio de hifens unitários ( - ) e/ou duplos ( -- ), ao passo que a execução do **argument** depende da ordem dos mesmos para a função.

```
$ ls --colors=auto
```

## { Comandos iniciais

Os comandos a seguir implementam operações básicas em arquivos:

- ls : Listar o conteúdo do diretório corrente ou de um diretório dado
- rm: Remove arquivos
- mv: Movimenta arquivos
- cp: Copia arquivos
- cat: Apresenta conteúdo de arquivos
- more: Visualiza conteúdo de arquivos (paginado)
- ln: Cria links (atalhos)

Os comandos usados para navegação na árvore de diretórios são similares àqueles usados em outros sistemas operacionais:

- pwd: Indica o diretório-corrente do Shell
- cd: Troca de diretório
- cd dir: Mudança para o diretório dir
- cd ..: Mudança para o diretório-pai imediatamente superior
- cd -: Volta para o último diretório visitado
- cd ~user: Leva para o diretório HOME do usuário indicado
- cd: Volta para o diretório HOME
- mkdir dir: Cria diretório dir
- rmdir dir: Remove o diretório dir
- chmod: Altera as permissões do diretório
- grep: Busca padrões especificados em um arquivo de texto



Os comandos que administram e processam operações diretamente do root são:

- sudo: Abreviação de *super user* do, é comumente usado para instalar software ou editar arquivos fora do diretório inicial do usuário

### Referências:

Comandos básicos:

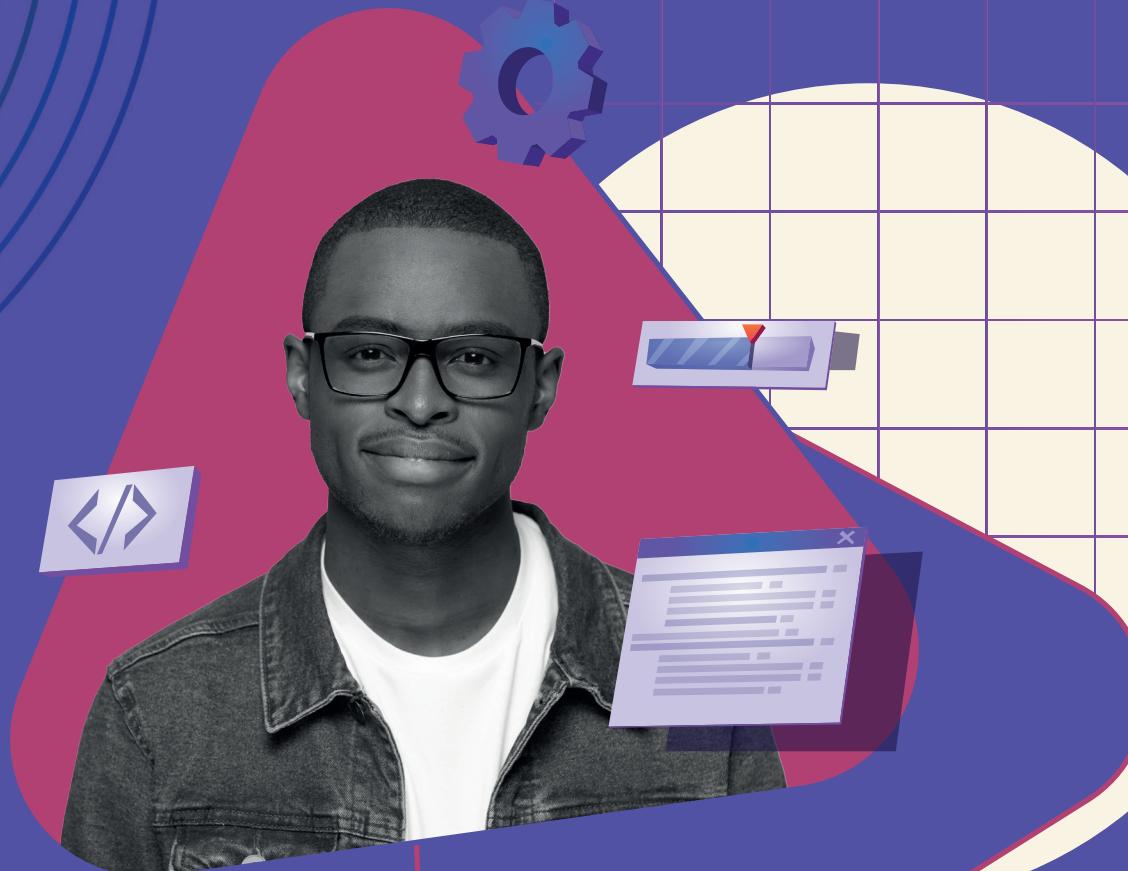
[http://wiki.inf.ufpr.br/maziero/doku.php?id=unix:comandos\\_basicos](http://wiki.inf.ufpr.br/maziero/doku.php?id=unix:comandos_basicos)

O que é CLI? <https://www.hostinger.com.br/tutoriais/o-que-e-cli>

O que é Unix? <https://luby.com.br/infraestrutura/o-que-e-unix/>

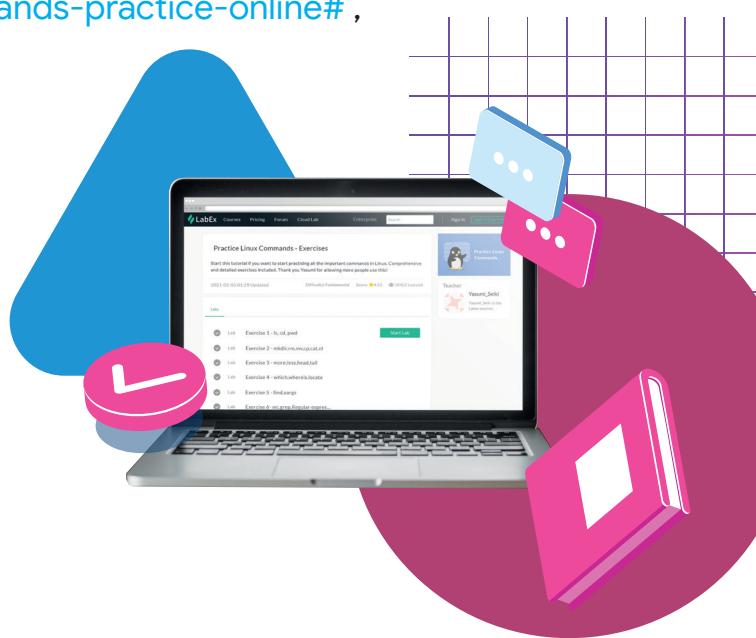
# <COMANDOS INICIAIS>

## /exercícios/



## { Introdução

Começamos pela prática no site <https://labex.io/courses/linux-basic-commands-practice-online#>, com diversos exercícios práticos.



## { Exercício coletivo I:

Liste todos os arquivos que contém no diretório principal, em seguida, liste os arquivos com as suas informações detalhadas.

## { Exercício coletivo II:

Navegue via linha de comandos até o diretório Documentos(Documents), criei um diretório com o nome projeto-service, crie um novo arquivo config.sh e utilize o comando específico para identificar o caminho inteiro do diretório onde foi criado o arquivo config.sh.

## { Exercício coletivo III:

*Utilizaremos o projeto-service como base para a resolução deste exercício.*

Siga o passo a passo:

- Crie um novo arquivo chamado dev.env, insira o seguinte parâmetro: *DB\_PASS: bd123*.
- Crie um novo arquivo chamado hom.env
- Copie a mensagem do arquivo dev.env para o arquivo hom.env
- Verifique se a mensagem está no arquivo hom.env
- Insira o parâmetro *COMMUNICATION\_S3 = cm123* no arquivo hom.env

## { Exercício reforço I:

*Utilizaremos o projeto-service como base para a resolução deste exercício.*

Crie um diretório dentro do projeto-service chamado user e um subdiretório chamado controller. Em seguida, volte para o diretório base - projeto-service.

## { Exercício reforço II:

*Utilizaremos o projeto-service como base para a resolução deste exercício.*

Acesse o projeto-service, verifique se o arquivo config.sh está na raiz do projeto e remova-o.

## { Sites para estudos e base para resolução dos exercícios:

- [https://github.com/krother/bash\\_tutorial](https://github.com/krother/bash_tutorial)
- [https://www.linuxpro.com.br/dl/guia\\_500\\_comandos\\_Linux.pdf](https://www.linuxpro.com.br/dl/guia_500_comandos_Linux.pdf)
- <https://ubuntu.com/tutorials/command-line-for-beginners#1-overview>

