

Bash, Prompt (CMD) e PowerShell (às vezes confundido com "prompt") são todos interpretadores de comandos, mas têm sintaxes, históricos e objetivos diferentes. Eles foram criados por empresas diferentes e refletem diferentes filosofias de sistema operacional.

Vamos por partes:

📌 Comparação direta: Bash vs CMD (Prompt) vs PowerShell

Característica	Bash	CMD (Prompt de Comando)	PowerShell
Desenvolvido por	GNU / Linux	Microsoft	Microsoft
Sistema operacional	Linux/macOS (Unix-like)	Windows	Windows (e depois Linux/macOS)
Sintaxe	Unix/Linux	Windows (MS-DOS-like)	Baseada em .NET (objetos)
Extensão de script	.sh	.bat ou .cmd	.ps1
Tipo de saída	Texto	Texto	Objetos estruturados
Supporte a scripts	Avançado	Limitado	Muito avançado
Acesso ao sistema	Forte (pipes, redirecionamento)	Básico	Muito forte (admin, rede, etc.)
Comum em servidores?	<input checked="" type="checkbox"/> Sim (Linux, cloud)	<input type="checkbox"/> Não	⚠️ Às vezes (ambientes Windows)
Comandos como ls, rm	<input checked="" type="checkbox"/> Sim	<input type="checkbox"/> Não (usa dir, del)	<input type="checkbox"/> (usa Get-ChildItem, Remove-Item)

🧠 Um pouco de contexto histórico:

🟩 Bash (Bourne Again Shell) – GNU/Linux

- Criado como parte do projeto GNU.

- É o shell mais comum em **Linux e macOS**.
 - Usa sintaxe Unix: ls, cd, rm, touch, chmod, etc.
 - Suporta **scripts poderosos**, automações, pipes (|), variáveis, loops, etc.
 - **Open-source**.
-

CMD (Command Prompt) – Microsoft

- Também conhecido como **Prompt de Comando** ou **Command Line**.
 - Herança do **MS-DOS** dos anos 80/90.
 - Usa comandos como dir, cd, copy, del, etc.
 - Muito limitado em comparação com Bash ou PowerShell.
 - Sintaxe mais simples, mas com **recursos limitados para scripts**.
-

PowerShell – Microsoft

- Criado para substituir o CMD.
 - Muito mais poderoso, voltado a administradores de sistemas.
 - Trabalha com **objetos**, não apenas texto.
 - Exemplo: comandos retornam objetos do .NET com propriedades manipuláveis.
 - Sintaxe é mais complexa (Get-Process, Set-ExecutionPolicy, etc).
 - Está disponível em **Windows, Linux e macOS** hoje em dia (open-source).
-

Exemplo prático: listar arquivos

Terminal Comando usado

Bash ls -la

CMD dir

PowerShell Get-ChildItem ou ls (alias)

Então, respondendo diretamente:

Eles têm sintaxes diferentes porque foram desenvolvidos por empresas diferentes com propósitos distintos.

- **Bash** → GNU/Linux (Unix-like)
 - **CMD** → Microsoft (MS-DOS)
 - **PowerShell** → Microsoft (plataforma moderna com .NET)
-

Qual usar?

Precisa de...	Use...
Scripts e automações em Linux/macOS	Bash
Comandos simples no Windows	CMD
Administração avançada no Windows	PowerShell
Scripts compatíveis com servidores	Bash
Terminal Unix-like no Windows	Bash via Git Bash ou WSL

Se quiser, posso te mostrar como alternar entre eles no VS Code ou escrever um script simples em cada um para comparação.

Você disse:

quais outros interpretadores de commandos existem, quantos são ?

O ChatGPT disse:

Excelente pergunta! Existem **diversos interpretadores de comandos (ou shells)**, criados ao longo do tempo para diferentes sistemas operacionais, linguagens ou finalidades. Alguns são amplamente usados; outros, mais de nicho.

O que é um interpretador de comandos (shell)?

É um programa que interpreta comandos digitados pelo usuário (ou lidos de um script) e os executa, geralmente interagindo com o sistema operacional.

Principais interpretadores de comandos por sistema

Sistemas Unix/Linux (baseados em POSIX)

Nome do Shell	Criado por	Descrição curta
sh (Bourne Shell)	AT&T (1979)	O shell original do Unix
bash	GNU (1989)	O mais usado em Linux; substitui o sh
zsh	Paul Falstad (1990)	Avançado e personalizável; usado no macOS
ksh (KornShell)	AT&T (1983)	Shell poderoso, mistura de sh e csh
csh (C Shell)	Berkeley (1978)	Sintaxe tipo C; menos usado hoje
tcsh	Berkeley (1981)	Extensão do csh com melhorias
fish	Fish shell team	Shell moderno, amigável, com sugestões automáticas
dash	Debian	Shell rápido e leve, compatível com POSIX

Sistemas Windows

Nome do Shell	Criado por	Descrição curta
CMD (Command Prompt)	Microsoft	Baseado no MS-DOS
PowerShell	Microsoft	Shell moderno baseado em .NET
Windows Terminal	Microsoft	Interface que agrupa outros shells (não é um shell por si só)
WSL	Microsoft	Permite rodar Bash e outros shells Linux no Windows
Git Bash	Git for Windows	Emula um terminal Bash no Windows
Cygwin	Red Hat	Ambiente Unix-like no Windows

Sistemas macOS

macOS é baseado em Unix, então ele suporta:

- bash (era o padrão até o macOS Catalina)
 - zsh (padrão atual desde 2019)
 - sh, ksh, csh, fish, entre outros (opcionais)
-

Quantos shells existem?

Depende de como você conta:

- **Shells principais e amplamente usados:** ~10–12
 - **Variedades e forks (derivações):** dezenas
 - **Total (incluindo experimentais, históricos ou de nicho):** mais de 30
-

Alguns outros menos comuns (mas interessantes):

Nome	Destaque
Elvish	Shell funcional e orientado a estrutura
NuShell (nushell)	Shell moderno com foco em estrutura de dados, tabelas
Xonsh	Shell baseado em Python
Oil Shell (OSH)	Reimplementação moderna do Bash/Sh
BusyBox shell (ash)	Shell superleve usado em sistemas embarcados
Rc	Shell do sistema Plan 9 (minimalista)

Critérios que diferenciam shells

- **Sintaxe (tipo C, POSIX, Python...)**
- **Funcionalidades interativas (autocompletar, sugestões)**
- **Programabilidade (scripts complexos, estruturas de controle)**
- **Compatibilidade com POSIX**
- **Portabilidade (Windows, Linux, etc.)**

Conclusão

Existem **muitos** interpretadores de comandos, mas os mais comuns e usados no dia a dia são:

- bash (Linux/macOS)
- zsh (macOS)
- sh (scripts POSIX compatíveis)
- CMD (Windows)
- PowerShell (Windows + cross-platform)
- fish (usuários avançados)
- WSL (para usar shells Linux no Windows)