Comandos do shell

O **shell**, interpretador de comandos, é um programa que recebe comandos pelo teclado e providencia a execução desses comandos.

Comandos são ordens passadas ao sistema operacional para executar uma determinada tarefa.

Para colocar o shell em execução no Ubuntu, você deve abrir a lista de aplicativos clicando no botão que fica no canto inferior esquerdo de sua tela. Procure pelo aplicativo **Terminal** e clique sobre ele. Será aberta uma janela de texto com uma linha contendo o nome de seu usuário, o caractere @, o nome de seu sistema, e a sequência **:~$**. Essa linha é conhecida como **prompt**.

Quando um usuário comum estiver utilizando o sistema, o shell apresenta, no final do prompt, o caractere **$**. Quando o shell estiver em uso pela conta root (administrador do sistema), o final do prompt será **#**.

Para executar um comando, é necessário que ele tenha permissão de execução e que esteja no caminho de procura de arquivos (esteja no **path**).

O **path** é o caminho de procura dos arquivos executáveis. Ele é armazenado na variável de ambiente **PATH**. Você pode ver o conteúdo desta variável com o comando:

```
echo $PATH
```

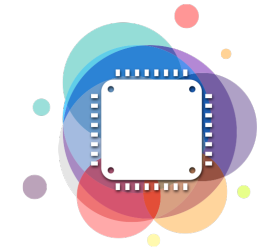
Um exemplo de retorno deste comando é:

```
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/
usr/games:/usr/local/games:/snap/bin
```

Por exemplo, o caminho "*/usr/local/sbin:/usr/local/bin:/usr/sbin*" significa que, ao ser digitado um comando, o interpretador de comandos iniciará a procura do comando no diretório "*/usr/local/sbin*". Caso não encontre o arquivo nesse diretório, procurará em "*/usr/local/bin*", depois em "*/usr/sbin*" e assim por diante. Caso o interpretador de comandos chegue até o último diretório do path e não encontre o arquivo digitado, será mostrada uma mensagem de erro.

Ao digitar o nome de um programa ou comando e teclar *Enter*, o programa/comando será executado e receberá um **número de identificação** (**PID – Process Identification**).

Todo o programa recebe também uma **identificação de usuário** (**UID – User Identification**) quando é executado, que determina quais serão suas permissões de acesso durante sua execução. O programa normalmente usa o UID do usuário que o executou, mas pode ser configurado pelo sistema para executar como um usuário específico.



Comandos podem possuir uma ou mais opções. Entre o nome do comando e suas opções deve haver, pelo menos, um espaço em branco. As **opções** são usadas para controlar como o comando será executado.

Comandos podem ser internos ou externos. **Comandos internos** são comandos que estão localizados dentro do interpretador de comandos (não ficam no disco). Eles são carregados na memória RAM do computador com o interpretador de comandos.

Quando executa um comando, o interpretador de comandos primeiramente verifica se é um comando interno. Caso não seja, é verificado se é um comando externo.

Comandos externos estão localizados no disco. Os comandos são procurados no disco de acordo com o **PATH** do sistema e são executados assim que encontrados.

É possível consultar a lista de comandos já executados utilizando as setas para cima e para baixo do teclado.

A tela pode ser rolada para frente e para trás ao pressionar a tecla **Shift** e clicando nas teclas **Page Up** e **Page Down**.

Programas podem executar em primeiro ou em segundo plano. Quando um programa está sendo executado em **primeiro plano** (foreground), é necessário esperar o término da execução do programa para executar outro. O prompt só é mostrado após o término da execução do programa.

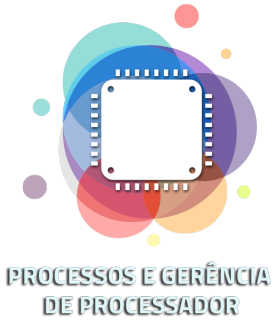
Quando um programa está sendo executado em **segundo plano** (background), não é necessário esperar o término de sua execução para executar um outro. Após iniciar um programa em background, é mostrado um **número PID** (identificação do processo) e o prompt é liberado, permitindo a execução de outro programa.

Para iniciar um programa em primeiro plano, basta digitar seu nome normalmente. Para iniciar um programa em segundo plano, acrescente o caractere **&** ao final do comando.

Teste a utilização do terminal:

- Execute o comando **gedit**.
- Tente utilizar o terminal enquanto a janela do gedit estiver aberta.
- Feche o gedit.
- Execute o comando "**gedit &**".
- Tente utilizar o terminal enquanto a janela do gedit estiver aberta.
- Feche o gedit.

Separando os comandos por "**;**" (ponto e vírgula), eles serão executados em sequência. Digite o comando "**gedit; xcalc; xeyes**" e tecle **Enter**.



Comandos para controle de processos

ps

Mostra os processos em execução no sistema, a identificação do usuário que executou o processo, a hora em que o processo foi iniciado etc.

Algumas opções:

- a → Processos de todos os usuários.

- x → Processos que não são controlados pelo terminal.
- u -> Mostra o nome de usuário que iniciou o processo e a hora em que o processo foi iniciado.
- m → Mostra a memória ocupada pelos processos.
- f → Mostra a árvore de execução de processos.

Exemplo: ps -aux

top

Mostra continuamente informações sobre processos, utilização da UCP, uso da memória RAM, swap etc.

Para encerrar a execução do top, deve ser pressionada a tecla **q**.

Algumas teclas úteis:

- espaço → Atualiza imediatamente a tela.

- h → Mostra a tela de ajuda.
- q -> Sai do programa.
- k -> Finaliza um processo. Semelhante ao comando kill. Será necessário fornecer o número de identificação do processo (PID).

<Ctrl> + <C>

Encerra o processo que está sendo executado em primeiro plano.

<Ctrl> + <Z>

Pausa a execução do processo que está em primeiro plano e mostra o número de seu job.

O prompt do shell é liberado.

O processo permanece na memória no ponto de processamento em que parou quando foi pausado. Sua execução pode ser retomada com os comandos fg ou bg.

jobs

Mostra os processos e seus respectivos números de job, que estão parados ou executando em segundo plano.

bg

Coloca um processo parado para executar em segundo plano. Para especificar o processo que entrará em execução em segundo plano, deve ser passado ao comando seu número obtido com o comando jobs.

fg

Coloca um processo parado ou executando em segundo plano para executar em primeiro plano. Para especificar o processo que irá para o primeiro plano, deve ser passado ao comando seu número obtido com o comando **jobs**. Caso não seja fornecido o número do job, ele irá para o primeiro plano o último processo pausado.

kill

Envia sinais a processos. Caso seja usado sem parâmetros, o kill enviará um sinal de término ao processo.

Comando: kill [opções] [sinal] [número]

Onde:

- número → Número de identificação do processo obtido com o comando ps. Também pode ser [%num], onde um é o número pelo comando jobs.

- sinal → Sinal que será enviado ao processo. Se omitido, envia o sinal 15 (SIGTERM).

Como opção pode ser passado ao comando a opção -9, que envia o sinal 9 (SIGKILL) ao processo. Trata-se um sinal de destruição que faz com que o processo seja terminado imediatamente.

Somente o dono do processo ou o usuário root pode terminá-lo ou destruí-lo.

killall

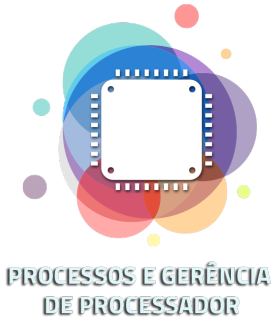
Finaliza um ou mais processos e tem seu nome como base.

killall5

Envia sinal de finalização a todos os processos.

nohup

Executa o comando passado como parâmetro, ignorando os sinais de interrupção.



Ainda pode ser finalizado com kill e <Ctrl>+<C>, mas sobrevive ao fechamento do terminal.

pstree

Mostra a árvore de processos do sistema.

Exemplos:

1) Encerrando processos com kill.

Abra um shell e coloque o gedit em execução com o comando:

```
gedit &
```

Execute o comando “ps -a” para listar os processos em execução e procure pela linha correspondente ao programa gedit. Um exemplo de saída do comando é:

```
  PID TTY          TIME CMD
 1495 tty2     00:00:55 Xorg
 1522 tty2     00:00:00 gnome-session-b
 3379 pts/0    00:00:02 gedit
 3394 pts/0    00:00:00 ps
```

Para o exemplo acima, percebemos que o gedit possui PID com valor 3379. O PID de seu processo deve ser outro. Verifique com atenção esse número.

Para eliminar o processo 3379, utilize o comando:

```
kill 3379
```

Repare que o processo gedit foi fechado.

2) Processos em primeiro plano e em segundo plano.

Abra um shell e coloque o gedit em execução com o comando:

```
xeyes
```

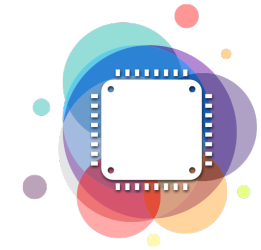
Será colocado em execução o aplicativo xeyes, um par de olhos que acompanham a posição do cursor do mouse.

Mexa o mouse pela tela e veja que o processo está funcionando normalmente.

Em seguida, tente digitar algum comando no terminal. Você não conseguirá, pois o processo xeyes está executando em primeiro plano, impedindo o acesso ao terminal.

Depois, pressione simultaneamente as teclas <Ctrl> e <Z>. Essa ação irá bloquear o processo xeyes e librar o prompt. A partir daí, você poderá entrar com novos comandos que o shell aceitará.

Processos bloqueados não realizam processamento. Mexa novamente o mouse e verá que os olhos não acompanham o cursor.



Para recolocar o processo em execução, você pode utilizar o comando fg (traz o processo para execução em primeiro plano) ou o comando bg (coloca o processo em execução em segundo plano). Vamos tentar com o bg. Para isso, execute o comando:

```
bg
```

Mexa o mouse e tente utilizar o comando. Você verá que ambos estão funcionando, pois, como o xeyes está executando em segundo plano, o terminal fica liberado.

Vamos manter o xeyes em execução.

Para colocar um processo em execução diretamente no segundo plano, você pode colocar o caractere & ao final do comando. Execute o comando:

```
xcalc &
```

A calculadora abrirá diretamente no segundo plano e teremos os três processos funcionando normalmente (shell, xeyes e calculadora).

Execute o comando:

```
gedit
```

Volte ao terminal e pressione <Ctrl>+<Z>. Com isso, o gedit será bloqueado e o shell liberado para execução.

Execute o comando:

```
jobs
```

Será apresentada uma saída como:

```
[1]   Executando                 xeyes &
[2]-  Executando                 xcalc &
[3]+  Parado                     gedit
```

Pela saída, podemos ver que os processos *xeyes* e *xcalc* estão executando em segundo plano e que o processo gedit está bloqueado. Você pode colocar qualquer um deles executando em segundo plano com o comando fg, bastar colocar o número do job (que aparece entre colchetes) como parâmetro para o comando.

Para colocar, por exemplo, o gedit executando em primeiro plano, execute o comando:

```
fg 3
```