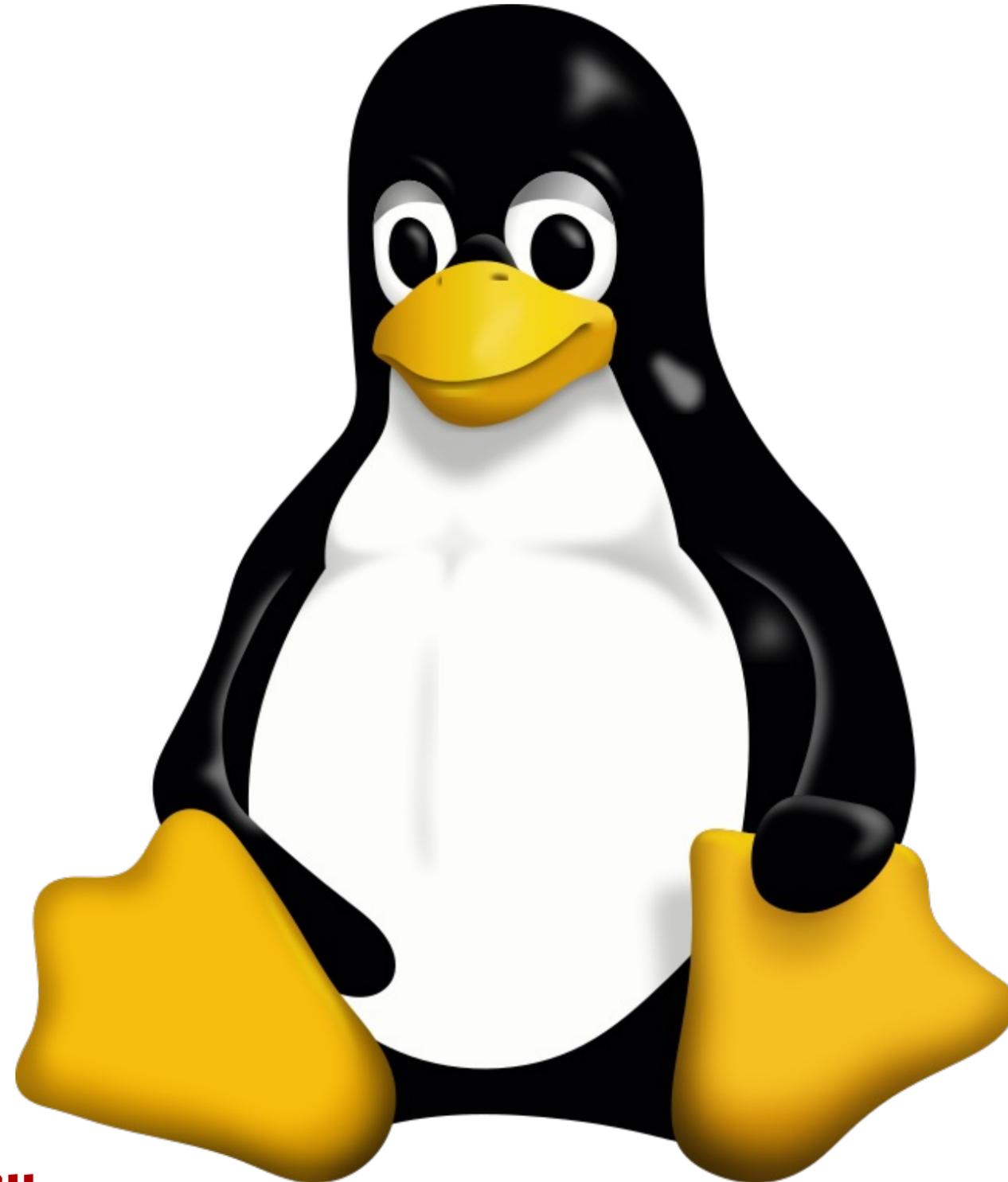


Sistemas Operacionais: navegação básica



O diretório de trabalho atual

Em qualquer momento do tempo você está dentro de um **diretório**, uma "pasta" no sistema de arquivos do Linux. No Windows costuma-se utilizar a nomenclatura de "pastas" mas, no Linux, o termo correto é diretório. Para ver qual é seu diretório de trabalho atual, use o comando:

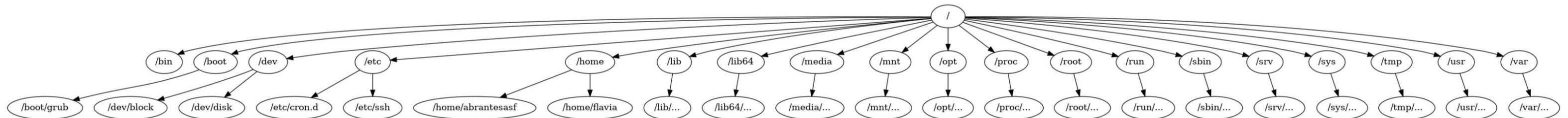
`pwd`

É nesse diretório que seus arquivos serão criados, apagados, etc. Em resumo, o diretório de trabalho é o diretório atual onde você está trabalhando na máquina.

```
abrantesasf@server01:~$ pwd  
/home/abrantesasf
```

Hierarquia de diretórios

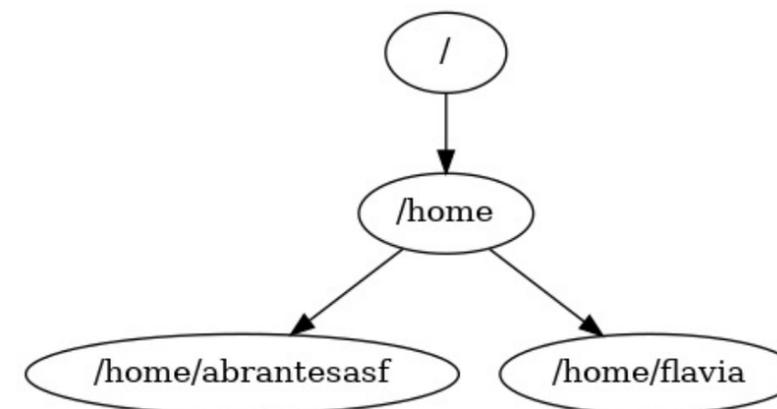
O Linux tem uma **hierarquia de diretórios**, com vários diretórios dentro de outros diretórios, formando uma espécie de uma árvore invertida de diretórios (pois a raiz está na parte de cima):



O diretório inicial da hierarquia é o diretório **raiz** (diretório **root**), representado apenas por uma barra: **/**

Todos os outros diretórios são "filhos" do diretório raiz. Por exemplo, meu diretório de trabalho pode ser:

/home/abranterasf

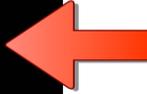


Qual o conteúdo do diretório?

Para listar o conteúdo (arquivos e outros diretórios) do diretório atual de trabalho, usamos o comando:

ls

```
abrantesasf@server01:~$ pwd
/home/abrantesasf
abrantesasf@server01:~$ ls
abrantesasf@server01:~$
```



Ao usarmos o comando **ls** no diretório **/home/abrantesasf**, não tivemos nenhum retorno, ou seja: o diretório está aparentemente vazio, sem outros arquivos ou subdiretórios. Na verdade há arquivos e diretórios ocultos (depois aprenderemos como mostrar esses objetos ocultos).

Como mudar de diretório?

Para alterar o diretório de trabalho, devemos usar o comando:

```
cd [pathname]
```

Os colchetes indicam que alguma coisa é opcional. Eles não devem ser digitados!

O comando **cd** significa "change directory". Para irmos até algum diretório, basta escrever o **pathname** (caminho) desse diretório. Mas atenção: há 2 tipos de **pathnames**: absoluto e relativo.

- **path absoluto**: sempre começa no diretório raiz
- **path relativo**: começa no diretório de trabalho atual

ATENÇÃO: todos os diretórios em um **pathname** devem ser obrigatoriamente separados por uma barra (/)

Como mudar de diretório? path absoluto

Vamos até o diretório **/** e ver seu conteúdo. Para isso vamos usar um path absoluto:



```

abrantesasf@server01:~$ cd /
abrantesasf@server01:/$ ls
bin          home          mnt          sbin.usr-is-merged  usr
bin.usr-is-merged  lib          opt          snap            var
boot         lib64         proc         srv
cdrom        lib.usr-is-merged  root         swap.img
dev          lost+found    run          sys
etc          media         sbin        tmp
abrantesasf@server01:/$ _
  
```

Note que o prompt mudou! Antes o path mostrava **~** e depois passou a mostrar **/**. O prompt sempre mostre o diretório atual de trabalho. Ao mudarmos para o diretório raiz, o prompt está mostrando isso.

Como mudar de diretório? path absoluto

Vamos até o diretório **/usr/local** e ver seu conteúdo. Para isso vamos continuar usando paths absolutos:

```
abrantesasf@server01:~$ cd /
abrantesasf@server01:/$ ls
bin          home          mnt          sbin.usr-is-merged  usr
bin.usr-is-merged  lib          opt          snap              var
boot         lib64         proc         srv
cdrom        lib.usr-is-merged  root        swap.img
dev          lost+found    run         sys
etc          media         sbin        tmp
abrantesasf@server01:/$ cd /usr/local ←
abrantesasf@server01:/usr/local$ ls
bin  etc  games  include  lib  man  sbin  share  src
abrantesasf@server01:/usr/local$ _
```

Note que o prompt mudou novamente! Agora ele mostra o diretório de trabalho **/usr/local**.

Como mudar de diretório? path relativo

Paths relativos nunca começam no diretório raiz, eles sempre são relativos ao diretório atual de trabalho. Para usarmos paths relativos temos de conhecer duas notações especiais:

- (ponto) diretório de trabalho**
- • (ponto ponto) diretório pai do diretório de trabalho**

É importante que você aprenda a trabalhar com eficiência com paths relativos!

Como mudar de diretório? path relativo

Vá até o diretório **`/usr/include/linux/cifs`**.

```
abrantesasf@server01:/usr/include/linux/cifs$ _
```

Como você iria para o diretório pai, o **`/usr/include/linux`**?

```
abrantesasf@server01:/usr/include/linux/cifs$ cd ..  
abrantesasf@server01:/usr/include/linux$
```



Usando a notação `..` (ponto ponto) podemos ir rapidamente para o diretório pai, sem ter que digitar novamente o path absoluto. Isso pode ser expandido para o diretório avô, bisavô, etc., assim:

```
abrantesasf@server01:/usr/include/linux$ cd ../../..  
abrantesasf@server01:/usr$ _
```

avô pai atual pai avô

Como mudar de diretório? path relativo

Vá até o diretório **/usr/include**.

```
abrantesasf@server01:/usr/include$
```

Como ir para o subdiretório **linux**, o **/usr/include/linux**?



```
abrantesasf@server01:/usr/include$ cd ./linux  
abrantesasf@server01:/usr/include/linux$ _
```

Usando a notação **.** (ponto) podemos ir rapidamente para um subdiretório filho do diretório atual, sem ter que digitar novamente o path absoluto. Isso pode ser expandido para o diretório neto, bisneto, etc., assim:

atual filho neto

```
abrantesasf@server01:/usr/include$ cd ./linux/cifs  
abrantesasf@server01:/usr/include/linux/cifs$ _
```

Como mudar de diretório? path relativo

Você pode usar combinações complexas de paths para especificar exatamente onde você quer ir na hierarquia de diretórios. Tente executar e entender o comando abaixo:

```
abrantesasf@server01:/usr/include/linux/cifs$ cd ../../x86_64-linux-gnu/sys/  
abrantesasf@server01:/usr/include/x86_64-linux-gnu/sys$
```

Como mudar de diretório? path relativo

Na prática, em praticamente todos os casos, se quisermos entrar em subdiretórios do diretório atual podemos omitir o **. /** pois ele é considerado automaticamente.

Fazer isso:

```
abrantesasf@server01:/usr$ cd ./bin  
abrantesasf@server01:/usr/bin$ _
```

É a mesma coisa de fazer isso:

```
abrantesasf@server01:/usr$ cd bin  
abrantesasf@server01:/usr/bin$ _
```

Na prática nunca usamos o **. /** e escrevemos direto o diretório filho.

Como mudar de diretório? opções úteis

O comando `cd` pode ser utilizado com diversas opções que fornecem facilidades na navegação entre os diretórios. Você pode usar:

- `cd` vai para seu diretório HOME
- `cd ~` vai para seu diretório HOME
- `cd -` vai para o último diretório visitado antes do atual
- `cd ~bob` vai para o diretório HOME do usuário bob
(se você tiver permissão para isso)

```
abrantesasf@server01:/usr/include/x86_64-linux-gnu/gnu$ cd  
abrantesasf@server01:~$ cd -  
/usr/include/x86_64-linux-gnu/gnu  
abrantesasf@server01:/usr/include/x86_64-linux-gnu/gnu$ cd ~  
abrantesasf@server01:~$ cd ~root  
-bash: cd: /root: Permission denied  
abrantesasf@server01:~$ _
```

O seu diretório HOME

Todos os usuários têm um diretório especial chamado de HOME. O usuário é o dono desse diretório, e tem permissão total sobre ele. É nesse diretório que o usuário pode criar arquivos, escrever documentos, apagar arquivos e diretórios, criar programas, criar scripts shell... enfim, é a "casa" do usuário no sistema.

Esse diretório HOME é representado pela notação `~` (til). Por isso fazer `cd ~` vai para o diretório HOME do usuário.

A informação de qual é o diretório HOME do seu usuário fica armazenada em uma **variável de ambiente no sistema**, que pode ser exibida com o comando `echo` da seguinte forma:

```
echo $HOME
```

```
abrantesasf@server01:~$ echo $HOME  
/home/abrantesasf
```

O `echo` é um comando que imprime alguma coisa na tela. `HOME` é a variável de ambiente. E o `$` significa o conteúdo da variável.

Informações importantes sobre arquivos

Nomes de arquivos que começam com um ponto, por exemplo, **.config**, ficam ocultos no sistema. Além disso, os nomes de arquivos, diretórios e comandos diferenciam letras maiúsculas de minúsculas. Assim, **arquivo.txt** é diferente de **Arquivo.txt**.

Não existe o conceito de extensão de arquivo como no Windows. Um **arquivo.doc** não significa nada, não é um documento do Word; um **arquivo.pdf** não significa nada, não é um documento PDF. Essas extensões no Linux não querem dizer nada. Nós usamos essas letras no nome para nos ajudar a lembrar o que é, mas não são obrigatórias! O conteúdo ou propósito de um arquivo é determinado por outros meios. *

* Apesar das extensões não serem usados no Linux, alguns programas e aplicativos podem fazer uso delas para melhorar a experiência do usuário.

Você pode escrever nomes de arquivos do jeito que quiser, mas para manter sua sanidade, utilize apenas letras sem acentos, pontos, hífen e underscores. E o mais importante: **não use espaços nos nomes de arquivos!** Você pode? Sim. Você deve? NUNCA!