

Curso de Linux

Módulo Básico



Sumário

Introdução	1
1 O que é Linux?	2
1.1 Um breve histórico	2
1.2 Software Livre e Licença GPL	3
1.3 Distribuições	3
1.3.1 Debian	4
1.3.2 Red Hat Enterprise Linux	4
1.3.3 Slackware	4
1.3.4 Ubuntu	4
2 Instalando	5
2.1 Como obter	5
2.2 Considerações sobre hardware	5
2.2.1 Configuração mínima	5
2.2.2 Configuração recomendada	5
2.3 Instalando o Ubuntu 9.10	6
3 Ambiente Gráfico	11
3.1 X Window System	11
3.2 Ambientes Desktop	11
3.3 Desempenhando tarefas	12
3.3.1 Acessando a internet	12
3.3.2 Editando um documento num processador de texto	13
3.3.3 Instalando programas	13
4 Aplicativos	15
5 Diretórios e arquivos	16
5.1 Visão geral da organização dos arquivos no Linux	16
5.1.1 Diretório root – /	16
5.1.2 /bin	17
5.1.3 /boot	17
5.1.4 /dev	17
5.1.5 /etc	17
5.1.6 /home	17
5.1.7 /lib	17
5.1.8 /media	17
5.1.9 /root	17
5.1.10 /tmp	17
5.1.11 /usr	17
5.1.11.1 /usr/bin	17
5.1.11.2 /usr/src	17
5.1.12 /var	18

5.1.12.1	/var/log	18
5.1.12.2	/var/run	18
5.2	Caminho absoluto X Caminho relativo	18
5.3	Permissões de acesso	18
5.3.1	Donos, grupos, outros	18
5.3.2	Tipos de permissões	19
6	Modo texto	20
6.1	Shell	21
6.2	BASH	21
6.3	Comandos	21
6.3.1	Prompt	21
6.3.2	Sintaxe dos comandos	21
6.3.3	pwd	22
6.3.4	ls	22
6.3.5	cd	24
6.3.6	mkdir	25
6.3.7	rmdir	26
6.3.8	touch	26
6.3.9	rm	27
6.3.10	cp	28
6.3.11	mv	29
6.3.12	cat	29
6.3.13	find	31
6.3.14	clear	31
6.3.15	exit	31
6.3.16	echo	31
6.3.17	date	32
6.3.18	chmod	32
6.3.19	passwd	34
6.3.20	su	34
6.3.21	sudo	35
6.3.22	wc	35
6.4	Pipe e redirecionamento	36
6.4.1	(Pipe)	36
6.4.2	>	37
6.4.3	>>	38
6.5	Instalando programas pela linha de comando	39
7	Obtendo ajuda	40
7.1	Comandos e opções	40
7.1.1	man	40
7.1.2	apropos	40
7.1.3	--help	41
7.2	Internet e literatura	41
7.3	Sugestões	41

Introdução

Este material destina-se a qualquer pessoa que queira adentrar ao curioso mundo do sistema operacional GNU/Linux tendo pouquíssimo ou nenhum conhecimento sobre o assunto.

O objetivo deste curso é que você, após concluir a leitura e prática de todo o conteúdo proposto, seja capaz de instalar e utilizar com um certo conforto o ambiente GNU/Linux.

O material está estruturado da seguinte maneira: o primeiro capítulo apenas introduz o GNU/Linux e o conceito de Software Livre. A seguir, mostramos como instalar um sistema GNU/Linux - você verá como pode ser incrivelmente fácil. Nos capítulos seguintes, fornecemos uma visão geral de como trabalhar no ambiente gráfico e no modo texto. Finalizamos com um capítulo que se propõe a mostrar como obter ajuda e ampliar seus conhecimentos.

Capítulo 1

O que é Linux?

O termo *Linux* é usado em vários contextos com significados diferentes. A rigor, Linux é um kernel. No entanto, em alguns contextos, Linux significa sistema operacional (não qualquer sistema operacional, mas um que use o kernel Linux).

Sistema Operacional: é um software que serve de interface entre o computador e o usuário, gerenciando recursos (como memória, processamento etc.).

Kernel: é o núcleo ou cerne do sistema operacional (é a parte deste que fica mais “próxima” do hardware).

Você pode agora estar se perguntando se deve chamar apenas o kernel de Linux. Como dito anteriormente, a rigor, Linux é o kernel. Contudo, a expressão “sistema operacional Linux” tornou-se muito difundida. Outra pergunta pode ter surgido neste ponto: qual o nome do sistema operacional então? Mais uma controvérsia aqui. Quando algum usuário instala “o Linux”, ele está instalando o kernel e mais uma série de outros softwares (aplicativos etc.). Grande parte desses aplicativos pertence a um projeto chamado GNU. Logo, o sistema operacional formado pelo kernel mais utilitários e aplicativos, como defendem alguns, deveria ser chamado de GNU/Linux.

1.1 Um breve histórico - Como surgiram o GNU e o Linux

No ano de 1984, Richard Stallman iniciou o Projeto GNU, que tinha por objetivo criar um sistema operacional que fosse totalmente livre. Esse sistema operacional deveria ser compatível com outro sistema operacional - o UNIX (daí o nome GNU - GNU is Not Unix). No ano seguinte, Stallman fundou a FSF (Free Software Foundation), com o propósito de eliminar restrições de uso, cópia e distribuição de software.

Por volta de 1991, o sistema GNU estava quase pronto, exceto pelo kernel. Stallman estava trabalhando no desenvolvimento de um kernel chamado Hurd. Ao mesmo tempo, o finlandês Linus Torvalds havia criado um kernel compatível com as aplicações do projeto GNU. A esse kernel foi dado o nome de Linux.

Atualmente, Linux tornou-se um termo genérico para se referir a sistemas operacionais “Unix-like” baseados no kernel Linux. Tornou-se, também, o melhor exemplo de Software Livre e de código aberto.



Figura 1.1: *Linus Torvalds*

1.2 Software Livre e Licença GPL

Na Seção anterior, foi dito que Stallman pretendia criar um sistema operacional livre e que o GNU/Linux era um exemplo de Software Livre. A definição de Software Livre, dada pela FSF é:

Um software é considerado livre se atende às seguintes liberdades:

- Executar o software com qualquer propósito (liberdade nº 0).
- Estudar o funcionamento do software e adaptá-lo às suas necessidades (liberdade nº 1).
- Redistribuir (inclusive vender) cópias do software (liberdade nº 2).
- Melhorar o programa e tornar as modificações públicas para que a comunidade inteira se beneficie da melhoria (liberdade nº 3).

Ao contrário do que as pessoas pensam, Software Livre (do inglês *Free Software*) não é sinônimo de gratuito. O que ocorre é uma confusão envolvendo a palavra “free” em inglês, que significa tanto gratuito como livre. Mas o sentido que Stallman queria dar era de “livre”. De qualquer forma, a maioria dos softwares livres é distribuída de forma gratuita.



Figura 1.2: *Richard Stallman*

Grande parte dos projetos de software livre (incluindo o GNU/Linux) é distribuída sob a GPL (*General Public License* - Licença Pública Geral), que é a licença idealizada por Stallman e que se baseia nas quatro liberdades citadas anteriormente. Com a garantia destas liberdades, a GPL permite que os programas sejam distribuídos e reaproveitados, mantendo, porém, os direitos do autor por forma a não permitir que essa informação seja usada de uma maneira que limite as liberdades originais.

1.3 Distribuições



Distribuições Linux (também chamadas Distribuições GNU/Linux ou simplesmente *distros*) consistem em “pacotes” de software baseados no kernel Linux que incluem determinados tipos de software para satisfazer as necessidades de um grupo específico de usuários, dando assim origem a versões domésticas, empresariais e para servidores.

Exemplos de Distribuições Linux: Ubuntu, Debian, Slackware, Fedora, Red Hat, Arch, Gentoo, Mandriva, openSUSE etc. Qual é a melhor distribuição é uma questão de necessidade e gosto.

Apresentamos a seguir uma breve descrição de algumas distros, para que você possa ter uma ideia de suas principais características.

1.3.1 Debian

A distro Debian (ou Debian GNU/Linux) é desenvolvida pelo Projeto Debian, um grupo de voluntários mantido por doações através da organização sem fins lucrativos Software in the Public Interest (SPI).

Debian baseia-se fortemente no projeto GNU e tem como principais características um alto compromisso com estabilidade e segurança bem como uma grande facilidade no que concerne à instalação de programas, através de um gerenciador de pacotes completo (dpkg) e sua interface (apt), utilizados amplamente em outras distribuições.

A última versão estável desta distro é 5.0.

1.3.2 Red Hat Enterprise Linux

Red Hat Enterprise Linux é uma distro criada pela empresa norte-americana Red Hat. O foco desta distribuição é o mercado corporativo, incluindo versões para servidores e para desktops.

O Red Hat Enterprise Linux não possui um ciclo de lançamentos fixo: a versão atual é a 5, mas o Red Hat Enterprise Linux 6 tem previsão de lançamento para o primeiro semestre de 2010.

1.3.3 Slackware

Simplicidade e estabilidade são duas características marcantes nesta distribuição. Muito comum em servidores, procura ser uma distribuição “leve”, praticamente sem enfeites e rápida, muito apreciada por usuários mais experientes.

Encontra-se atualmente na versão Slackware 13.

1.3.4 Ubuntu

Ubuntu é uma distro GNU/Linux baseada na distro Debian e é patrocinada pela Canonical.

A proposta do Ubuntu é oferecer um sistema operacional que qualquer pessoa possa utilizar sem dificuldades, independentemente de nacionalidade, nível de conhecimento ou limitações físicas (a palavra Ubuntu é de origem africana e significa “humanidade para os outros”).

Essa distro oferece um ambiente atualizado e estável, focado na usabilidade e na facilidade de sua instalação.

A cada seis meses, uma nova versão da distro é lançada, a versão atual é Ubuntu 10.4. Os números 10 e 4 são, respectivamente, o ano e o mês do lançamento da versão.

Capítulo 2

Instalando

Este capítulo mostrará como instalar o Ubuntu 9.10 (32 bits) através de um CD-ROM. De fato, esta é uma tarefa muito simples, pois, ao longo dos anos, os instaladores de quase todas as distros tornaram-se bastante amigáveis, mesmo para usuários totalmente inexperientes.

2.1 Como obter

O Ubuntu pode ser obtido gratuitamente através do site <http://www.ubuntu.com/getubuntu>.

2.2 Considerações sobre hardware

Antes de proceder com a instalação, o usuário deve saber se sua configuração de hardware irá funcionar. Todas as distros publicam alguma lista de compatibilidade de hardware, consulte-a se estiver em dúvida sobre algum dispositivo.

Seguem abaixo as configurações mínima e recomendada para instalação em desktop do Ubuntu 9.10. É importante salientar que, devido ao ambiente gráfico, configurações superiores podem ser necessárias.

2.2.1 Configuração mínima

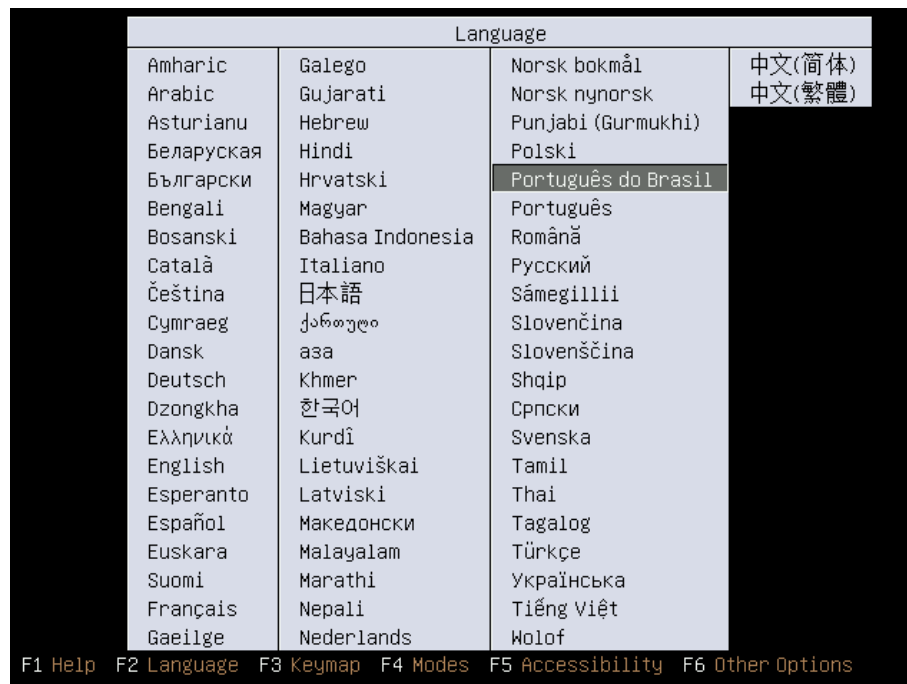
- processador 300 MHz x86
- 64 MB de RAM
- Pelo menos 4 GB de espaço em disco (para instalação completa e swap)
- placa de vídeo VGA com suporte para resolução 640x480
- drive CD-ROM ou placa de rede

2.2.2 Configuração recomendada

- processador 700 MHz x86
- 384 MB de RAM
- 8 GB de espaço em disco
- placa de vídeo com suporte para resolução 1024x768
- placa de som
- conexão com a Internet

2.3 Instalando o Ubuntu 9.10

1. A primeira tela da instalação deve ser esta que aparece abaixo. Selecione a língua de sua preferência.

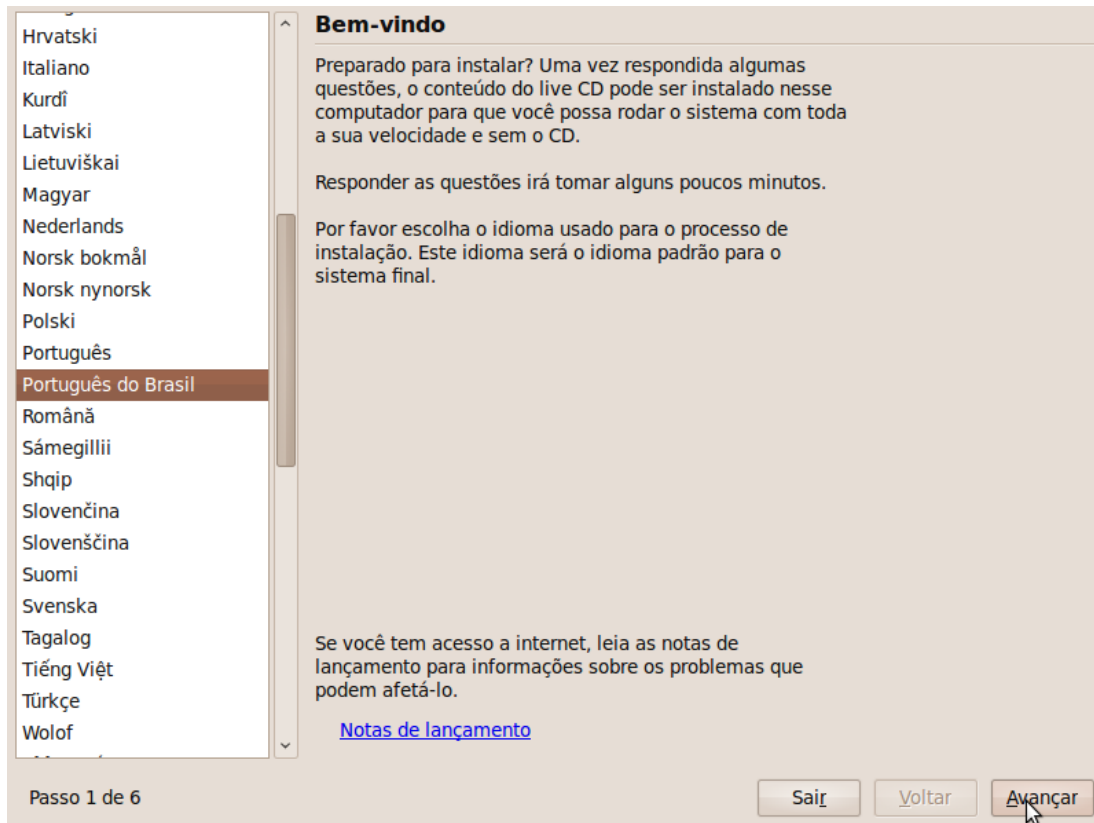


2. A seguir, selecione a segunda opção “Instalar o Ubuntu”.



Observação: O CD do Ubuntu é um Live CD. Isso significa que o usuário pode executar o sistema operacional direto do CD, sem precisar instalar nada nem efetuar qualquer mudança em seu disco rígido. Para fazer isso, basta selecionar a opção “Testar o Ubuntu sem qualquer mudança no seu computador”. Esta é uma boa alternativa para quem quer testar o sistema antes de instalá-lo.

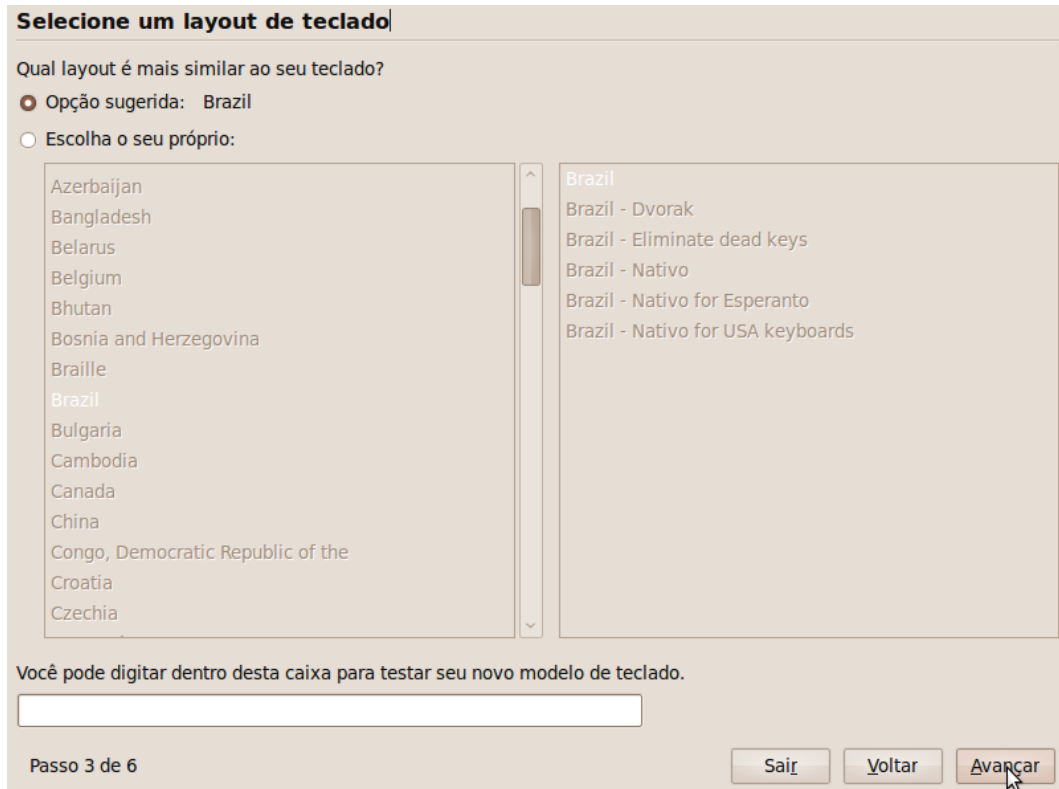
3. Você verá um wallpaper por alguns segundos. Quando o instalador aparecer, você poderá selecionar a língua de sua preferência para o processo de instalação e para o sistema.



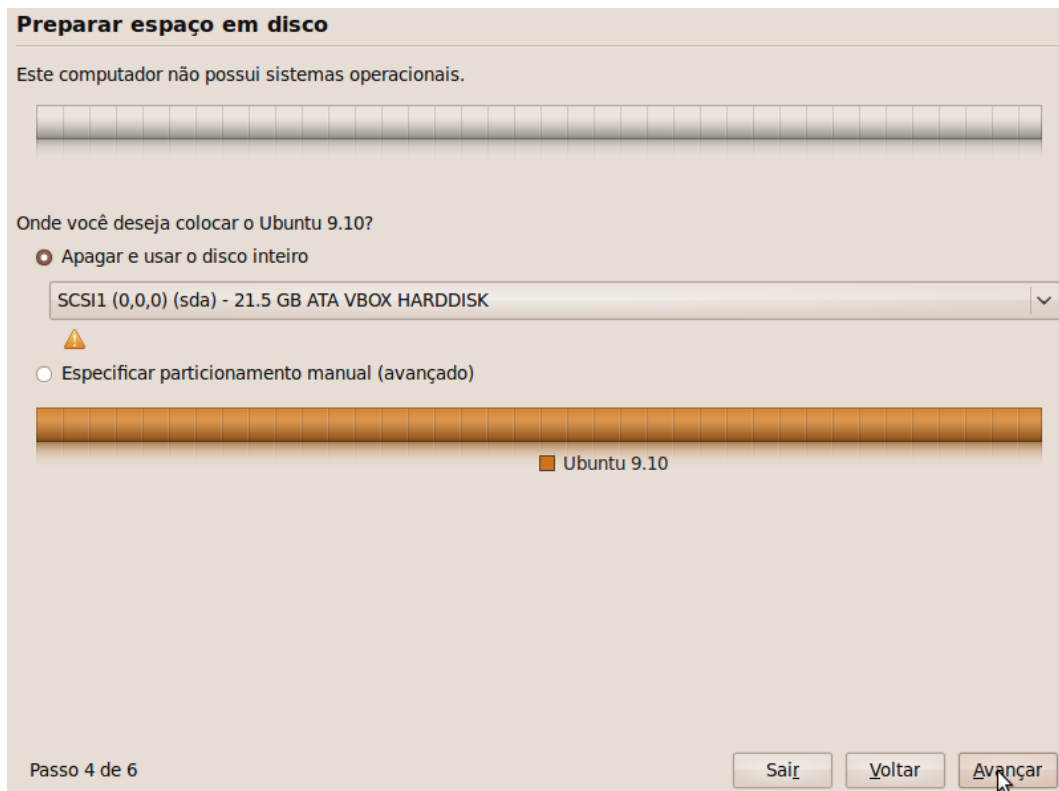
4. Agora você deverá selecionar sua localização, para que o horário seja ajustado pelo sistema e as atualizações sejam feitas a partir de locais mais próximos. Você poderá fazer isso clicando no mapa ou nas listas.



5. Selecione o layout do teclado.



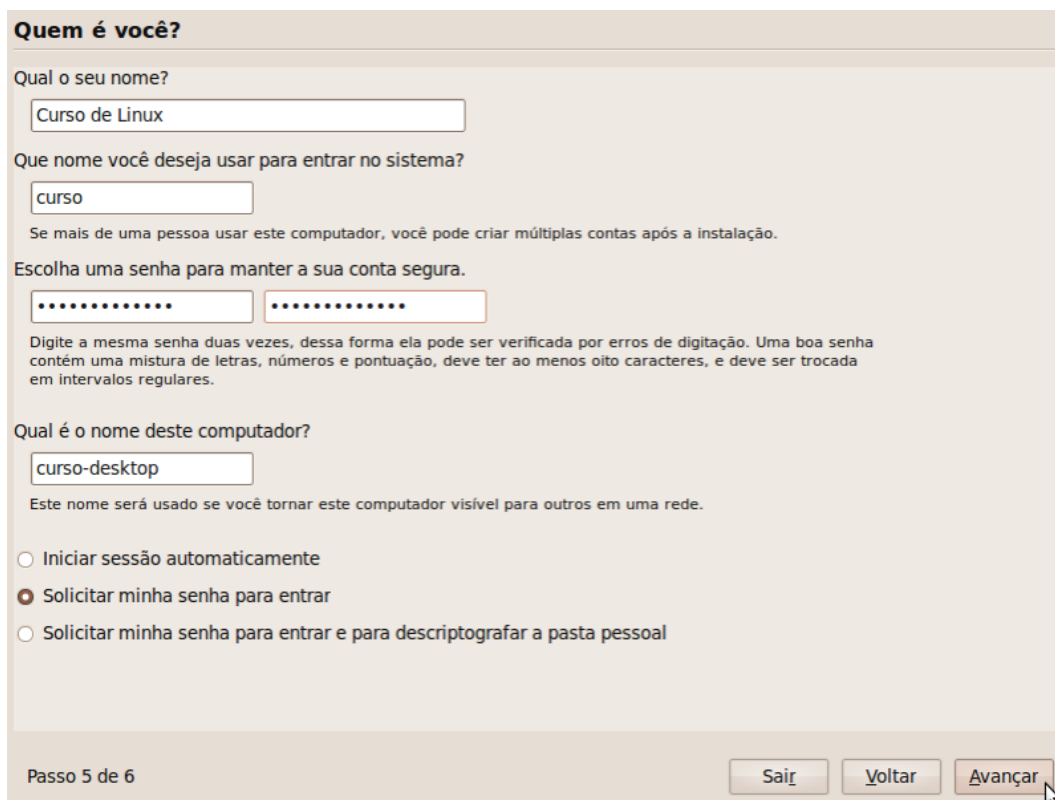
6. Esta é a parte em que o usuário irá decidir “onde” instalar o sistema. Neste módulo, não abordaremos todos os detalhes dessa etapa. Por enquanto, apenas mostramos como instalar o Ubuntu utilizando o disco todo.



Observação: É possível instalar o Ubuntu (e outros distros também) lado a lado com outros sistemas operacionais (incluindo outros distros). Isso significa que você não precisará abandonar o Windows (ou outro sistema de sua preferência) para poder instalar o GNU/Linux

em seu HD. É possível selecionar qual sistema se deseja usar no processo de boot da máquina. Este assunto será abordado num outro módulo do curso.

7. A seguir, preencha a tela seguinte, de acordo com o que cada título diz. Preencha com seu nome, com o nome que você deseja logar-se no Ubuntu (seu “username”), a senha de sua preferência e o nome do computador.



Quem é você?

Qual o seu nome?

Que nome você deseja usar para entrar no sistema?

Se mais de uma pessoa usar este computador, você pode criar múltiplas contas após a instalação.

Escolha uma senha para manter a sua conta segura.

Digite a mesma senha duas vezes, dessa forma ela pode ser verificada por erros de digitação. Uma boa senha contém uma mistura de letras, números e pontuação, deve ter ao menos oito caracteres, e deve ser trocada em intervalos regulares.

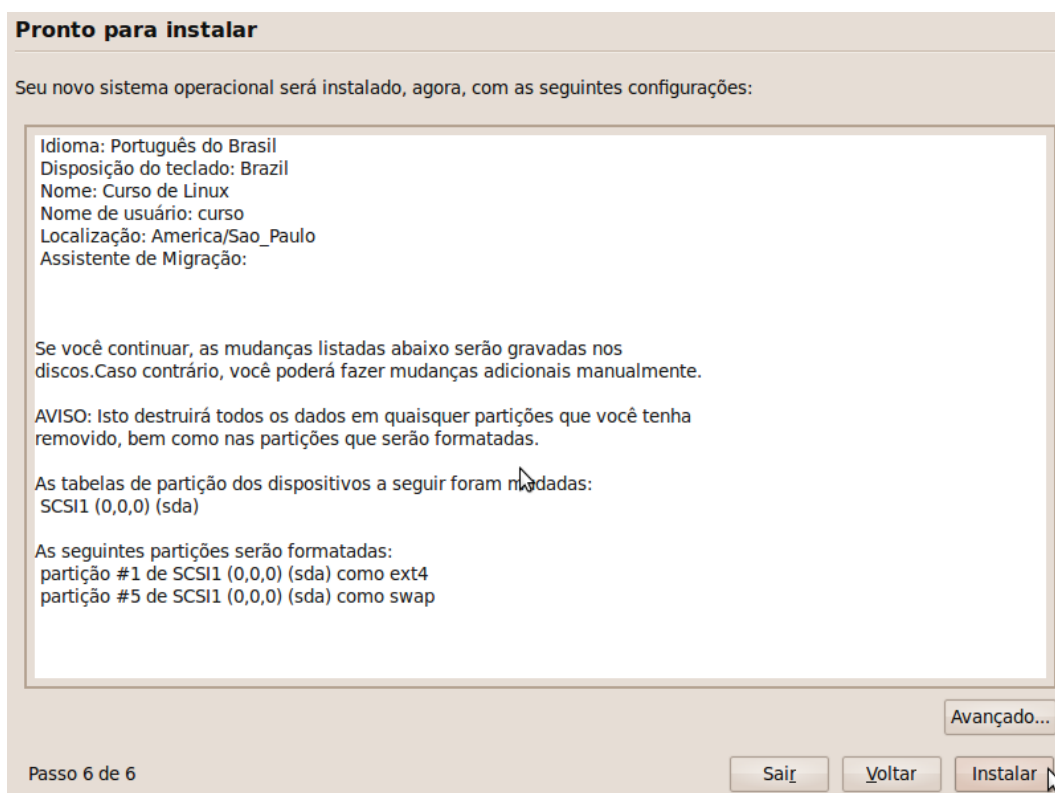
Qual é o nome deste computador?

Este nome será usado se você tornar este computador visível para outros em uma rede.

Iniciar sessão automaticamente
 Solicitar minha senha para entrar
 Solicitar minha senha para entrar e para descriptografar a pasta pessoal

Passo 5 de 6

8. Nesta tela, você deverá conferir se as opções definidas para a instalação estão corretas. Se estiver tudo ok, clique em “Instalar”.



Pronto para instalar

Seu novo sistema operacional será instalado, agora, com as seguintes configurações:

Idioma: Português do Brasil
Disposição do teclado: Brazil
Nome: Curso de Linux
Nome de usuário: curso
Localização: America/Sao_Paulo
Assistente de Migração:

Se você continuar, as mudanças listadas abaixo serão gravadas nos discos. Caso contrário, você poderá fazer mudanças adicionais manualmente.

AVISO: Isto destruirá todos os dados em quaisquer partições que você tenha removido, bem como nas partições que serão formatadas.

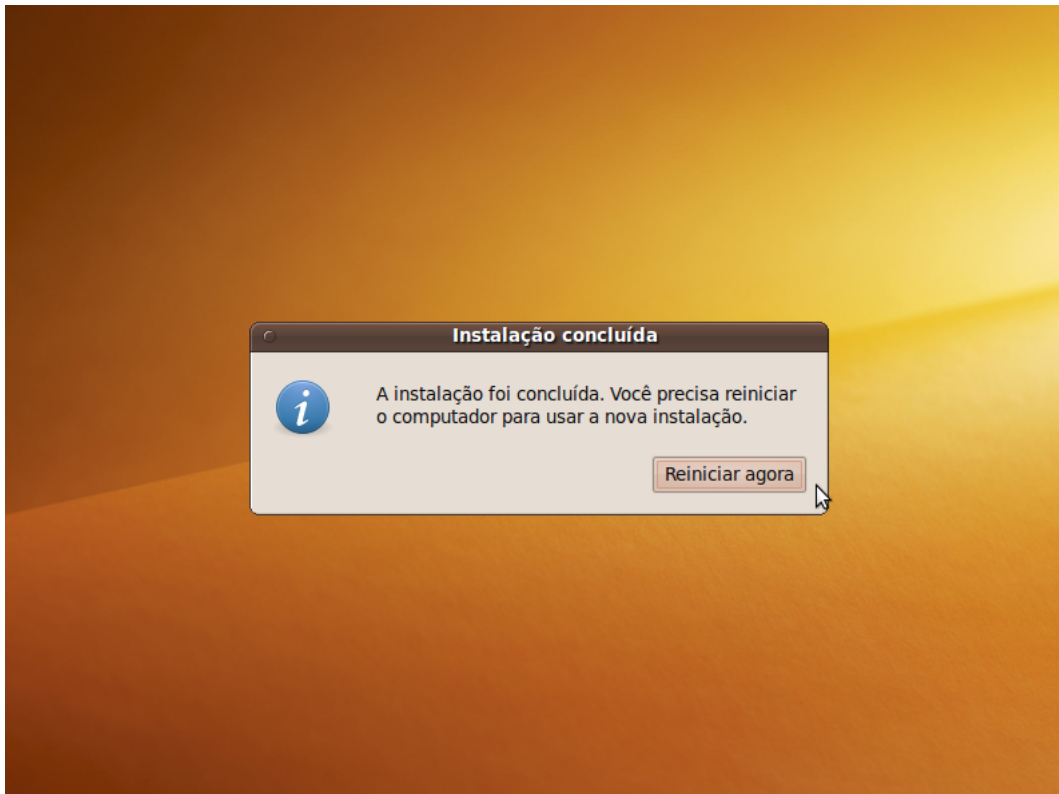
As tabelas de partição dos dispositivos a seguir foram mudadas:
SCSI1 (0,0,0) (sda)

As seguintes partições serão formatadas:
partição #1 de SCSI1 (0,0,0) (sda) como ext4
partição #5 de SCSI1 (0,0,0) (sda) como swap

Avançado...

Passo 6 de 6

9. O sistema será instalado, podendo levar de 10 a 20 minutos (a depender do hardware de sua máquina). Quando a instalação estiver completa, será necessário reiniciar o computador.



10. Pronto. O Ubuntu 9.10 está instalado em sua máquina.

Capítulo 3

Ambiente Gráfico

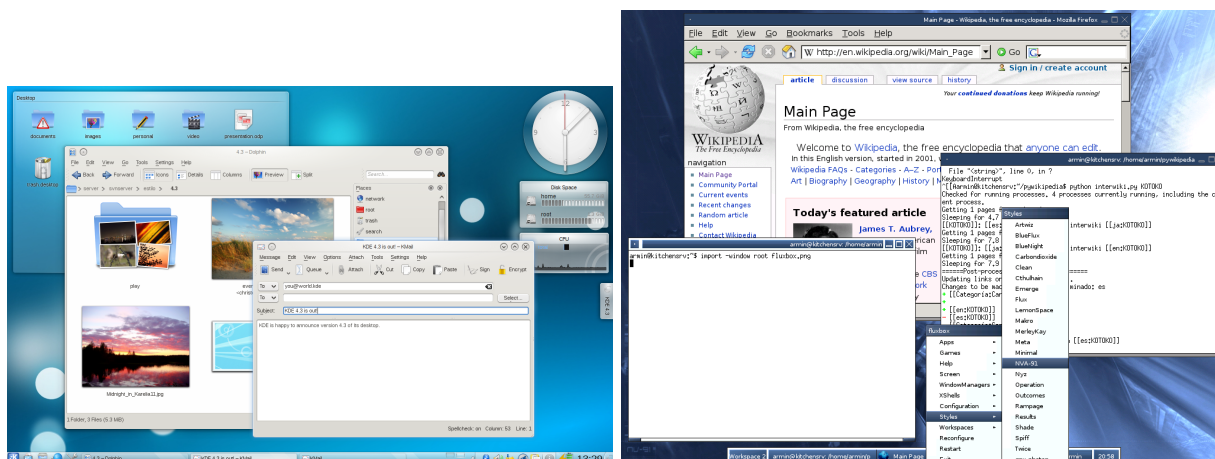
No Linux (como UNIX, em geral), o sistema operacional é independente da GUI (*Graphical User Interface* - Interface Gráfica do Usuário, ou simplesmente, interface gráfica). Existem várias vantagens que essa abordagem proporciona: permite que o usuário escolha a interface de sua preferência; além disso, se a interface gráfica falha, o sistema continua funcionando. Para alguns, pode ser um pouco estranho falar em sistema sem interface gráfica. Contudo, não é somente através do modo gráfico que o usuário consegue interagir com o sistema, é possível fazê-lo através de uma outra maneira - usando o modo texto, assunto que será abordado no Capítulo 6.

3.1 X Window System

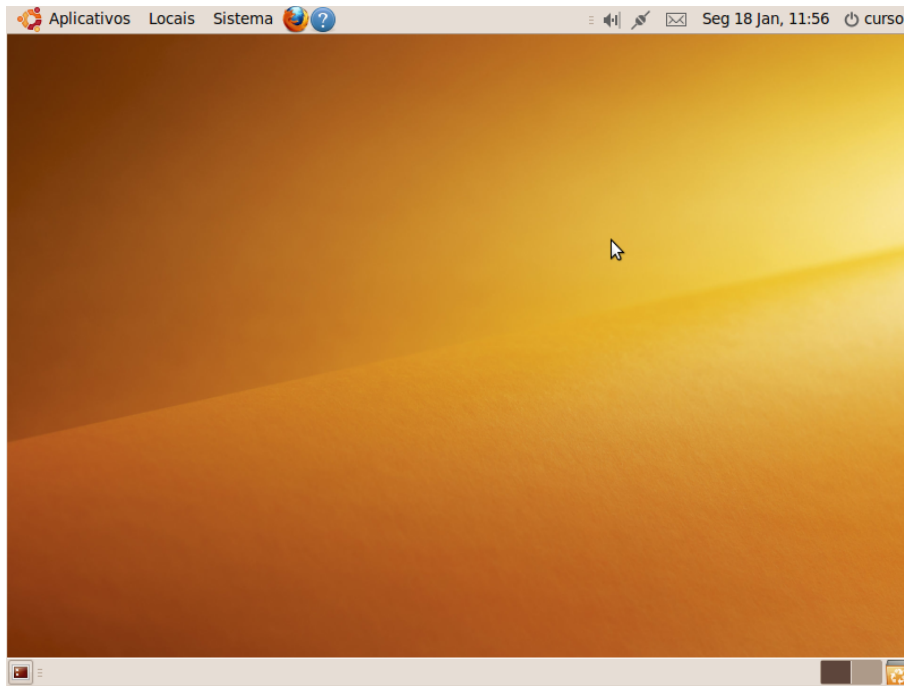
X Window System (ou simplesmente X11 ou X) é o toolkit e protocolo padrão para interface gráfica em plataformas UNIX e similares (como o Linux). Ele não é uma interface gráfica completa: apenas define como os objetos básicos devem ser desenhados e manipulados na tela. O X pode ser executado em máquinas locais ou remotamente, através de uma rede.

3.2 Ambientes Desktop

O X é a base para a interface gráfica no GNU/Linux, mas não inclui nenhuma definição sobre a aparência das janelas. O controle da aparência é feito por outro programa, chamado gerenciador de janelas (*window manager*). Exemplos de gerenciadores de janelas são o FVWM, FluxBox, Xfce, GNOME e KDE. Estes dois últimos transcendem o conceito de gerenciador de janelas e são ambientes desktop (*desktop environment*), ou seja, propõem-se a oferecer uma interface mais completa para o sistema operacional, incluindo utilitários integrados e aplicações.



KDE e FluxBox.



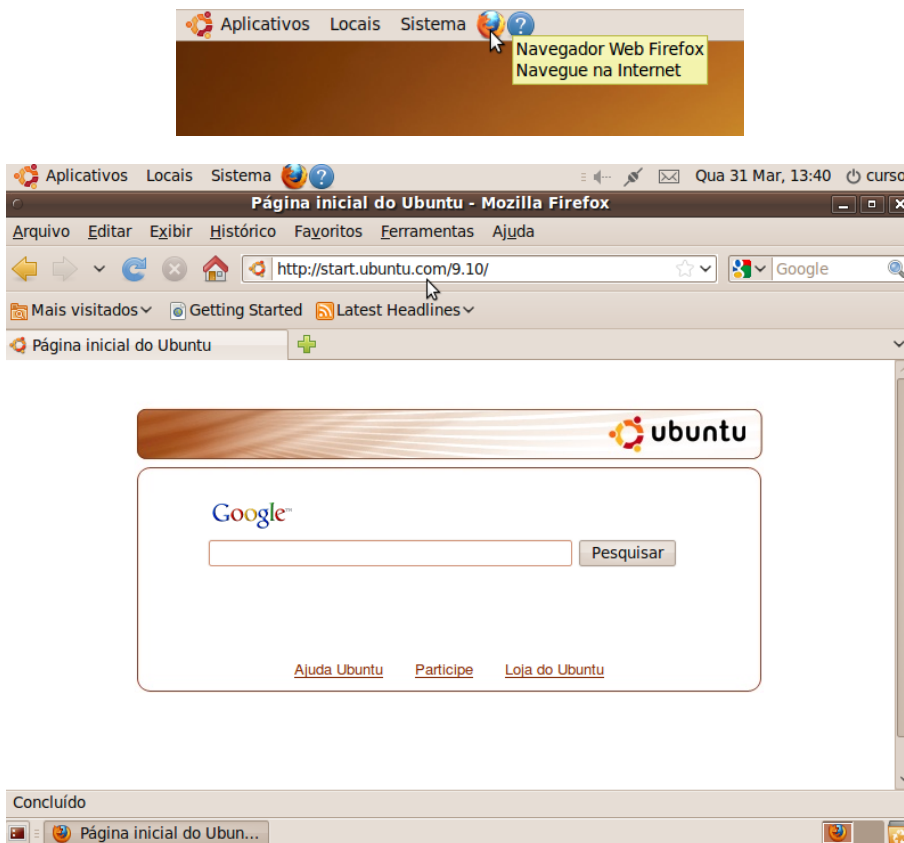
GNOME (Ubuntu 9.10).

3.3 Desempenhando tarefas

A proposta da interface gráfica é o uso intuitivo. Apesar disso, mostramos a seguir como desempenhar algumas tarefas simples e corriqueiras, apenas para efeito ilustrativo. A interface usada é o GNOME (Ubuntu 9.10).

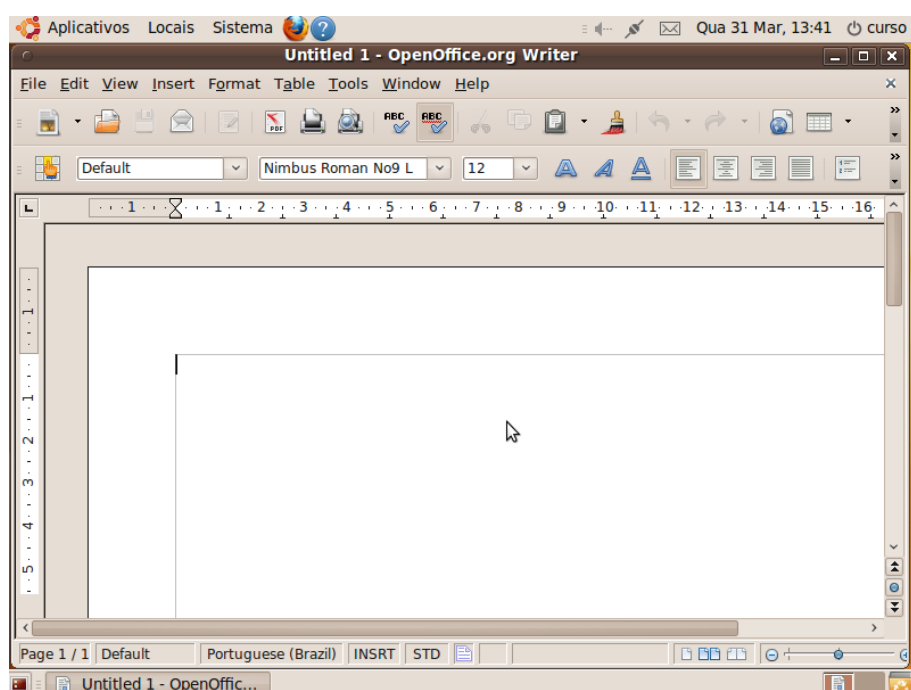
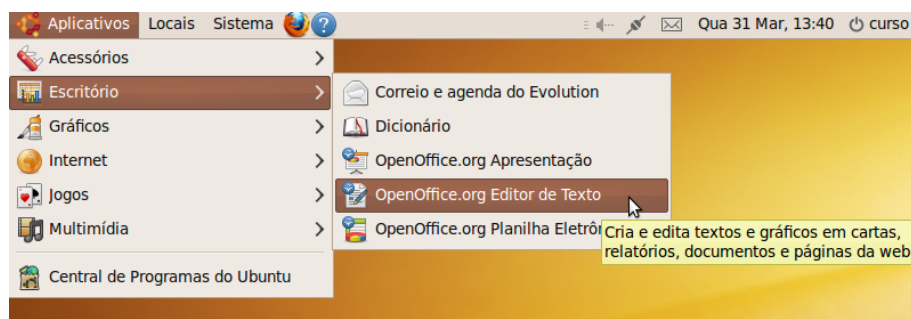
3.3.1 Acessando a internet

No Ubuntu 9.10, o navegador Firefox já vem instalado e já existe um atalho para acessá-lo:



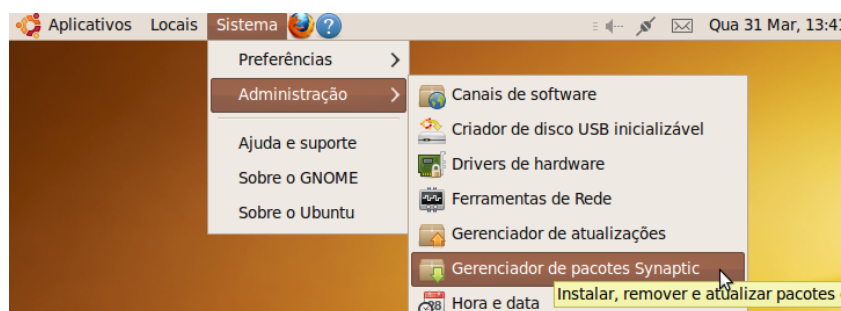
3.3.2 Editando um documento num processador de texto

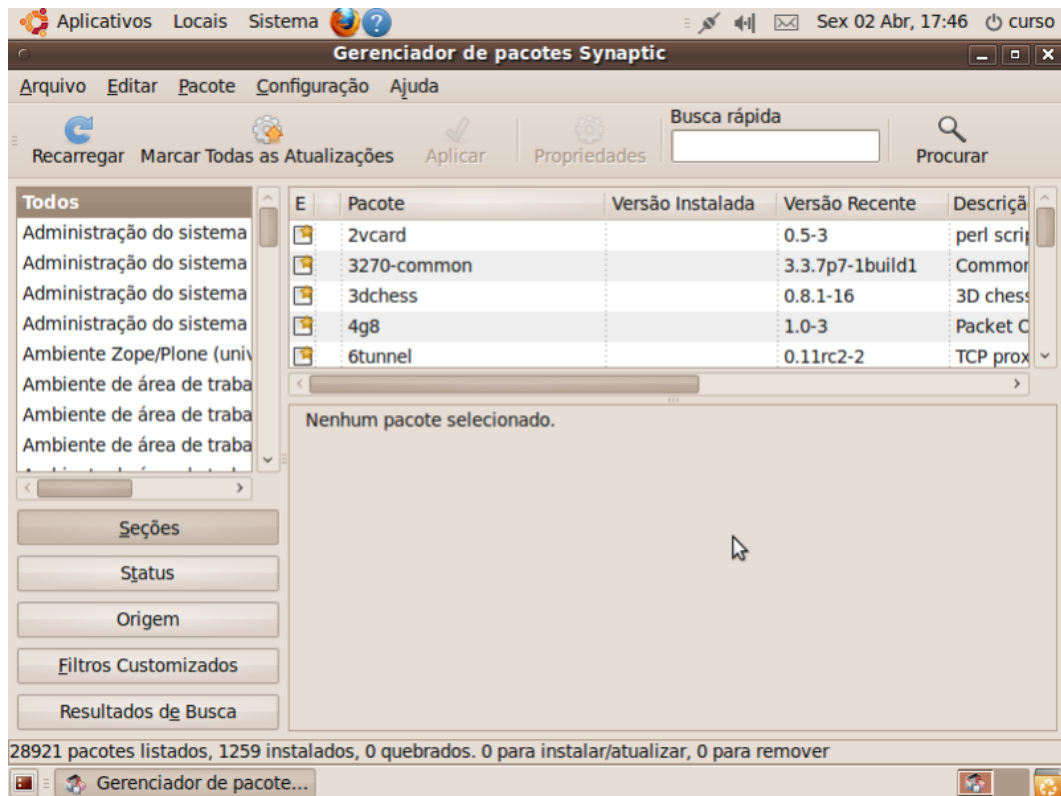
A maioria dos leitores deve estar acostumada a usar o Microsoft Word (do pacote Office) para editar arquivos de texto com formatação. No Ubuntu 9.10, já vem instalado o pacote do OpenOffice.org, que possui formatos próprios de arquivo mas que também consegue abrir arquivos “.doc”.



3.3.3 Instalando programas

Existem várias maneiras de instalar programas no GNU/Linux. Mostramos aqui uma ferramenta chamada Synaptic, presente não apenas no Ubuntu, mas em outras distros também. O Synaptic nada mais é do que um programa que oferece uma interface gráfica amigável para instalar programas.





Capítulo 4

Aplicativos

Basicamente, para qualquer programa que você utilizava no Windows, existe uma alternativa no GNU/Linux. A tabela abaixo propõe-se a oferecer algumas destas alternativas.

Descrição	Programas usados no Windows	Programas usados no GNU/Linux
Pacote Office	Microsoft Office	KOffice, OpenOffice
Processador de Texto	Microsoft Word	KWrite, OpenOffice Writer
Planilhas	Excel	KSpread, OpenOffice Calc
Apresentações	PowerPoint	KPresenter, OpenOffice Impress
E-mail	Outlook	Evolution
Gravação de mídia	Nero	Brasero, K3b
IDEs LaTeX	TeXnicCenter, WinEdit	Kile, Texmaker
Compactadores de arquivos	Winrar, Winzip	ark, bzip2, tar
Leitor de PDF	Adobe Reader	Adobe Reader, Evince, Kpdf
Modelagem 3D	3D Studio MAX, Blender, Maya	Blender, K-3D, Maya
Players de vídeo	Windows Media Player	Kaffeine, MPlayer Totem, VLC
Players de música	iTunes, Winamp, Windows Media Player	Amarok, Audacious, RhythmBox
Edição de vídeos	Windows Movie Maker	Cinelerra, Kino
Clientes P2P BitTorrent	μ Torrent, Azureus	Azureus, KTorrent, Transmission
Mensageiros instantâneos	MSN	aMSN, Kopete, Pidgin
Browser	Firefox, Google Chrome, Microsoft Internet Explorer, Opera	Firefox, Galeon, Google Chrome, Konqueror, Opera

Observação: Vários dos aplicativos listados apenas em “Programas usados no GNU/Linux” também funcionam no Windows (o VLC, por exemplo).

Capítulo 5

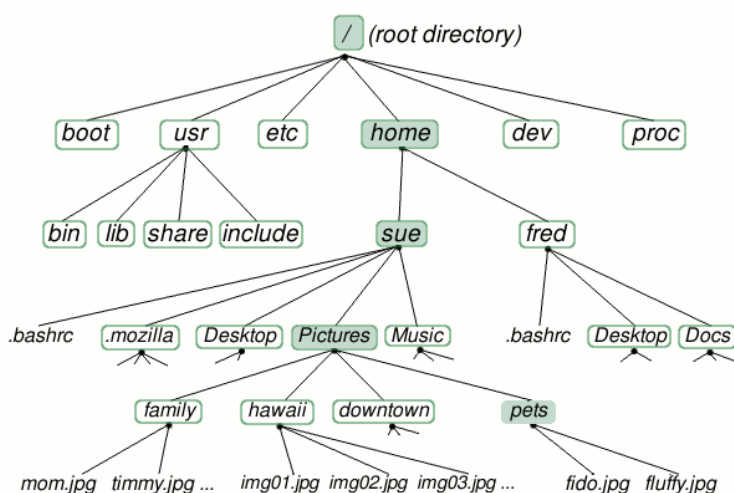
Diretórios e arquivos

Muitos usuários têm dificuldades com o GNU/Linux porque não têm uma visão geral sobre o que está guardado em que local. Neste capítulo, mostramos um pouco da organização dos arquivos do GNU/Linux.

5.1 Visão geral da organização dos arquivos no Linux

Grosso modo, pode-se dizer que, no Linux, tudo é arquivo. Se há algo que não seja um arquivo, então este algo é um processo. No GNU/Linux (como no UNIX), não há diferença entre arquivo e diretório, uma vez que um diretório é apenas um arquivo contendo nomes de outros arquivos. Imagens, músicas, textos, programas, serviços e assim por diante são todos arquivos. Dispositivos de entrada e saída, e geralmente, todos os dispositivos, são considerados como arquivos.

Todos estes arquivos estão organizados de acordo com uma hierarquia, isto é, há critérios que prevêm os principais diretórios e seu conteúdo. Estes critérios são definidos por um padrão, o FHS (*Filesystem Hierarchy Standard*).



No topo da hierarquia de arquivos fica o chamado diretório raiz (ou, mais apropriadamente, diretório root), pois a estrutura de diretórios é chamada também de “Árvore de Diretórios”.

5.1.1 Diretório root – /

Este é o diretório principal do sistema. Dentro dele estão todos os diretórios do sistema. O diretório root é representado por uma barra (/).

5.1.2 /bin

Contém comandos e programas essenciais para todos os usuários (alguns desses comandos serão tratados no próximo capítulo).

5.1.3 /boot

Contém arquivos necessários para a inicialização do sistema.

5.1.4 /dev

Dispositivos: o /dev contém referências para todos os dispositivos, os quais são representados como arquivos com propriedades especiais.

5.1.5 /etc

Contém arquivos de configuração.

5.1.6 /home

Contém os diretórios dos usuários.

5.1.7 /lib

Contém bibliotecas (que são subprogramas ou códigos auxiliares utilizados por programas) essenciais para o funcionamento do Linux, e também os módulos do kernel.

5.1.8 /media

Este diretório contém subdiretórios que são usados como pontos de montagem para mídias removíveis, como disquetes, cdroms, pen drives etc.

5.1.9 /root

Diretório “home” do super usuário (usuário root). **Não confundir com o diretório root, o /. O diretório /root contém os arquivos do usuário root. O diretório / é o topo da hierarquia de arquivos.**

Usuário root: É o administrador do sistema, possui acesso a todos os comandos e arquivos.

5.1.10 /tmp

Para arquivos temporários.

5.1.11 /usr

Contém programas, bibliotecas etc.

5.1.11.1 /usr/bin

É onde ficam os binários de programas não-essenciais (os essenciais ficam no /bin).

5.1.11.2 /usr/src

Código-fonte.

5.1.12 /var

Contém arquivos “variáveis”, como logs, base de dados.

5.1.12.1 /var/log

Como o próprio nome diz, possui arquivos de log.

Arquivo de log: É um arquivo que armazena registros de eventos relevantes de um programa ou do sistema.

5.1.12.2 /var/run

Contém informação sobre a execução do sistema desde a sua última inicialização.

Existem outros diretórios previstos no padrão, mas, por enquanto, estes já são suficientes.

5.2 Caminho absoluto X Caminho relativo

Caminho de um diretório são os diretórios que devemos percorrer até chegar a ele. Vamos diferenciar caminho absoluto de caminho relativo por meio de um exemplo.

Consideremos o diretório xinit (você não precisa se preocupar com ele, é apenas um exemplo. O que importa realmente aqui é o conceito de caminho até o arquivo). Consideremos que este diretório encontra-se dentro de um outro diretório, o diretório X11. Este X11, por sua vez, está dentro do diretório etc, que, finalmente, está sob o diretório root, o /. Recapitulando: temos o / e dentro o etc (/etc), e dentro o X11 (/etc/X11) que contém o xinit (/etc/X11/xinit). Logo, “/etc/X11/xinit” é o **caminho absoluto** para o diretório xinit, ou seja, são os diretórios que devemos percorrer, começando pelo /, até o diretório xinit.

Consideremos agora os mesmos diretórios do caso anterior (/etc/X11/xinit). Suponhamos agora que estamos no diretório etc. Para dizer qual é o caminho do diretório xinit, bastaria dizer apenas X11/xinit - este é o **caminho relativo** do diretório (em relação ao diretório /etc). Se estivéssemos no diretório X11, o **caminho relativo** seria simplesmente xinit.

Em suma, caminho absoluto é aquele que utiliza toda a estrutura de diretórios, ao passo que o relativo toma um diretório como referência e define o caminho a partir daí.

5.3 Permissões de acesso

O Linux foi desenvolvido para ser um sistema *multi-usuário*. Isto significa que vários usuários podem ter configurações personalizadas, independentes das dos demais usuários, bem como diferentes usuários podem executar tarefas ao mesmo tempo numa mesma máquina. Assim sendo, cada usuário pode querer negar ou permitir o acesso a determinado arquivo ou diretório. Por isso, existem as chamadas permissões de acesso do Linux: para impedir o acesso indevido de outros usuários ou mesmo de programas mal intencionados a arquivos e diretórios.

Mostraremos algumas destas permissões nesta seção. Mais adiante, no próximo capítulo, mostraremos como manipulá-las.

5.3.1 Donos, grupos, outros

No Linux, **para cada arquivo são definidas permissões para três tipos de usuários:** o dono do arquivo, um grupo de usuários e os demais usuários (que não são nem o dono, nem pertencem ao grupo).

- **Dono:** O dono do arquivo é o usuário que criou o mesmo. Somente o dono e o usuário root podem mudar as permissões para um arquivo ou diretório.
- **Grupo:** É um conjunto de usuários. Grupos foram criados para permitir que vários usuários tivessem acesso a um mesmo arquivo.
- **Outros:** Como dito anteriormente, são os usuários que não se encaixam nos tipos de usuários supracitados.

5.3.2 Tipos de permissões

Os três tipos básicos de permissão para arquivos e diretórios são:

- **r (read):** permissão de leitura para arquivos. Caso seja um diretório, permite listar seu conteúdo (com o comando `ls`, por exemplo - que será visto no próximo capítulo).
- **w (write):** permissão de escrita para arquivos. Caso seja um diretório, permite a gravação de arquivos ou outros diretórios dentro dele. Para que um arquivo/diretório possa ser apagado, é necessário o acesso à escrita (gravação).
- **x (execute):** permite executar um arquivo. Caso seja um diretório, permite que seja acessado através do comando `cd` (você verá este comando também no próximo capítulo, equivale a “entrar” no diretório).

Em suma, para cada arquivo do sistema, são definidas permissões para o dono do arquivo, para um grupo de usuários e para os demais usuários. Essas permissões são de leitura, escrita e execução (r, w ou x). Você entenderá melhor estes conceitos no próximo capítulo, mas tente familiarizar-se com eles desde já.

Capítulo 6

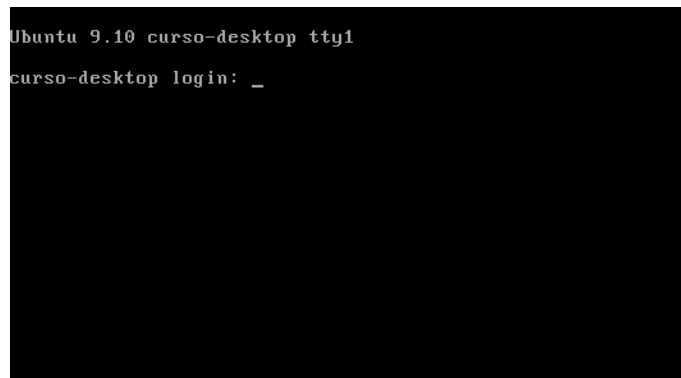
Modo texto

Como dito anteriormente, não é apenas pelo modo gráfico que o usuário consegue interagir com o sistema. É possível fazer isso pelo modo texto, digitando comandos e nomes de programas para conseguir uma “resposta” do sistema. Por isso, o modo texto é também chamado de linha de comando.

É importante para um usuário do GNU/Linux aprender a trabalhar no modo texto por vários motivos: otimiza várias tarefas, existem alguns programas que rodam somente no modo texto e também porque o modo gráfico consome mais recursos.

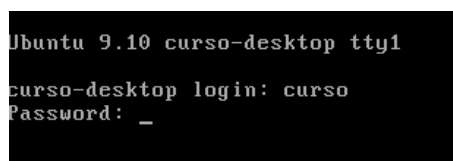
Você deve estar se perguntando agora como é que se faz para usar o GNU/Linux em modo texto. Na verdade, existem duas formas.

Você pode acessar um terminal “puro”, pressionando as teclas “Ctrl+Alt+F1” (substituir o F1 por F2, de F3 até F6 também funciona na maior parte das distros) e depois voltar ao modo gráfico pressionando “Alt+F7” (funciona para a maioria das distros).



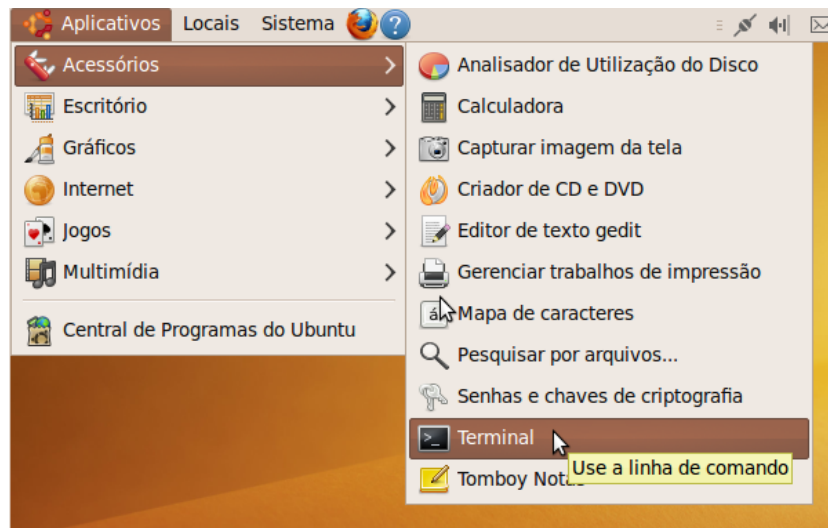
```
Ubuntu 9.10 curso-desktop tty1
curso-desktop login: _
```

Uma tela de login do modo texto geralmente mostra alguma informação sobre a máquina na qual você está trabalhando, o nome da máquina e um prompt para login. Para logar-se, digite o nome de usuário e tecla Enter. Agora você deverá digitar sua senha e teclar Enter novamente. O usuário não verá nenhuma indicação de que está digitando a senha (não aparecerão asteriscos nem nada do gênero, por motivos de segurança), mas isso é normal no GNU/Linux.



```
Ubuntu 9.10 curso-desktop tty1
curso-desktop login: curso
Password: _
```

A segunda forma é usar um “emulador de terminal”, isto é, dentro do modo gráfico, abre-se um programa que funciona como linha de comando. Para fazer isso no ambiente GNOME, vá em Aplicativos ⇒ Acessórios ⇒ Terminal.



6.1 Shell

De qualquer uma das duas formas, o que você verá rodando (após logar-se ou acessar o Terminal) é um programa chamado shell, que é um interpretador de comandos.

6.2 BASH

O BASH (Bourne Again Shell) é o shell desenvolvido para o projeto GNU, da Free Software Foundation, que se tornou padrão nas várias distribuições Linux (incluindo Ubuntu).

6.3 Comandos

Nesta seção, examinaremos alguns comandos simples do BASH. É importante que você saiba que não é preciso decorar os comandos apresentados. Para aprendê-los de fato, você deve ir praticando com os exercícios propostos e conforme a sua necessidade.

6.3.1 Prompt

O prompt do BASH tem a seguinte aparência:

```
username@nomedamáquina:diretório$
```

No caso de

```
curso@curso-desktop:~$
```

curso é o nome do usuário, curso-desktop é o nome da máquina, ~ é o diretório em que o usuário se encontra (~ representa o diretório home do usuário, nesse caso, /home/curso), e o \$ é o símbolo do tipo de usuário (nesse caso, um usuário normal). Se fosse o usuário root (administrador do sistema), o símbolo seria #.

6.3.2 Sintaxe dos comandos

É importante lembrar que a linha de comando é *case sensitive*, isto é, diferencia letras maiúsculas de minúsculas. Portanto, “echo” é diferente de “Echo”, que são diferentes de “ECHO”. Isso também vale para nomes de diretórios e arquivos.

Os comandos são, em geral, em letras minúsculas. Muitos deles aceitam argumentos. Os argumentos que começam com um (ou dois) “-” são opções.

```
comando -opção1 -opção2 --opção3 argumento
```


Quando os argumentos forem arquivos ou diretórios, tanto o caminho absoluto como o relativo poderão ser usados.

Outro ponto importante é que você pode digitar os comandos e nomes de arquivos ou diretórios pela metade e depois pressionar “Tab”. O shell “tentará completar” o que falta para você. Se houver mais de uma opção para completar o que foi digitado, as alternativas possíveis serão mostradas. Este é um recurso que facilita muito o uso da linha de comando.

Vamos então aos comandos.

6.3.3 pwd (print working directory)

Mostra o nome e o caminho do diretório atual (diretório em que o usuário está).

```
curso@curso-desktop:~$ pwd
/home/curso
```

6.3.4 ls (list)

Lista os arquivos e subdiretórios de um ou mais diretórios.

Sintaxe básica:

```
ls [opções] [diretório1] [diretório2] ...
```

Exemplos

1. O comando abaixo lista os diretórios e arquivos do /.

```
$ ls /
```

2. O comando abaixo lista os diretórios e arquivos do /etc.

```
$ ls /etc
```

3. Para listar o conteúdo do / e do /etc, de uma só vez, use:

```
$ ls / /etc
```

Exercício: Liste o conteúdo do diretório /tmp.

Para listar o conteúdo do diretório atual, basta digitar apenas “ls”. Se o usuário estiver em seu diretório home e digitar ls, a saída será os arquivos e diretórios contidos no /home/username.

Suponha ainda que o usuário encontra-se em seu diretório home. Existe, dentro do home do usuário, um diretório chamado “Documentos”. Se quisermos listar o conteúdo deste, podemos usar o comando

```
$ ls /home/username/Documentos
```

mas também podemos usar o caminho relativo (lembra-se?):

```
$ ls Documentos
```

Opções:

- **-a ou -all:** Lista todos os arquivos e diretórios, incluindo os ocultos. No GNU/Linux, os arquivos e diretórios ocultos começam por “.”. Quando usamos o comando `ls` como anteriormente (sem nenhuma opção), esses arquivos não são listados.

Exemplo

O comando abaixo listará todos os arquivos e diretórios contidos no barra, incluindo os ocultos.

```
$ ls -a /
```

Exercício: Liste todo o conteúdo do seu diretório `home`, incluindo os itens ocultos. (Quando fizer isso, você notará que dois itens “estranhos” foram listados: o “.” e o “..”. Eles representam, respectivamente, o diretório atual e o diretório acima. Se você estiver em seu diretório `home` e usar o comando “`ls ../`”, o conteúdo do `/home` será listado).

- **-R:** Lista o conteúdo de um diretório e dos subdiretórios, recursivamente. Quando você utiliza o comando `ls`, os arquivos e diretórios contidos num determinado diretório são mostrados. Usando a opção `-R`, serão listados os arquivos contidos num determinado diretório, e para cada subdiretório também serão listados os arquivos e diretórios nele contidos. E para cada um desses diretórios, também será listado todo o seu conteúdo e assim sucessivamente. Se você usasse “`ls -R /`”, o conteúdo de todos os diretórios seria mostrado (não estamos recomendando que você rode este comando, está aqui apenas para que você entenda o que faz esta opção).
- **-l:** Usa o formato longo para listagem, o que significa que serão listados detalhes sobre cada arquivo e diretório mostrado. Vamos examinar que detalhes são estes.

```
curso@curso-desktop:~$ ls -l
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Área de Trabalho
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Documentos
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Downloads
-rw-r--r-- 1 curso curso  167 2010-01-18 11:41 examples.desktop
-rw-r--r-- 1 curso curso    8 2010-01-18 12:24 exemplo
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Imagens
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Modelos
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Música
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Público
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Vídeos
```

Tomemos a primeira linha do resultado obtido:

```
drwxr-xr-x 2 curso curso 4096 2010-01-18 11:54 Área de Trabalho
```

- `drwxr-xr-x` – indicam as permissões.
- `2` – indica o número de subdiretórios contidos.
- `curso` – é o dono do arquivo ou diretório.
- `curso` – é o grupo ao qual o arquivo ou diretório pertence.
- `4096` – tamanho do arquivo (em bytes).
- `2010-01-18 11:54` – data e hora em que o arquivo ou diretório foi criado/modificado.
- `Área de Trabalho` – nome do arquivo ou diretório.

Permissões

A primeira letra (d) indica que “Área de Trabalho” é um diretório. Se fosse um arquivo normal, teríamos um “-” no lugar (é o caso de `examples.desktop` e `exemplo`). Os próximos nove caracteres representam as permissões do diretório. As permissões de um arquivo ou diretório são r, w e x, apresentadas no capítulo anterior (lembra-se? Leitura, escrita e execução.).

Para cada três caracteres são mostradas as permissões para um tipo de usuário. Os três primeiros caracteres, no caso “rwx”, indicam as permissões para o dono do arquivo. A interpretação desta trinca é que o dono do arquivo (no caso, o usuário “curso”), possui as três permissões sobre o diretório (leitura, escrita e execução). Os três próximos caracteres mostram as permissões para o grupo: “r-x”, o que significa que o grupo possui permissão de leitura e execução, mas não possui permissão de escrita (há um “-” no lugar do “w” de escrita). Por último, temos a permissão para os demais usuários do sistema (“r-x” – permissão de leitura e execução).

Número de subdiretórios

O número da segunda coluna representa o número de subdiretórios contidos. Se for um arquivo comum, esse número será 1.

Dono do arquivo

A terceira coluna representa o dono do arquivo, que, como dito anteriormente, é o usuário que criou o arquivo ou diretório.

Grupo

O grupo ao qual o arquivo pertence está mostrado na quarta coluna.

Tamanho

A coluna seguinte mostra o tamanho do arquivo em bytes. No caso de um diretório, não é mostrado o tamanho total, isto é, considerando todo o conteúdo do diretório, mas sim o tamanho da estrutura diretório, isto é, ainda que seja criado um diretório vazio, ele ocupará 4096 bytes de espaço em disco.

Como último comentário sobre este comando, vale dizer que é possível usar mais de uma opção de cada vez. Aliás, isso vale para todo comando.

O comando a seguir lista todos os diretório e arquivos do /, incluindo os ocultos, usando o formato longo de listagem.

```
$ ls -a -l /
```

Também é possível fazer isso da seguinte forma:

```
$ls -al /
```

Exercício: Liste os arquivos e diretórios do seu diretório home, incluindo os ocultos e o conteúdo dos subdiretórios.

6.3.5 cd (change directory)

Entra em um diretório.

Sintaxe básica:

```
cd [diretório]
```

Exemplos

1. Para entrar no diretório root, use

```
$ cd /
```

2. Para entrar no diretório /tmp, basta usar o seguinte comando

```
$ cd /tmp
```

3. Para subir um diretório acima, use:

```
$ cd ..
```

4. Para voltar ao diretório imediatamente anteriormente acessado, basta usar:

```
$ cd -
```

Exercícios

1. Entre no diretório home do seu usuário (“/home/seu-usuario-aqui”). Agora use o seguinte comando:

```
$ cd ../../
```

Use outro comando para descobrir em que diretório você acabou de entrar.

2. O que acontece se você digitar apenas o comando “cd”, sem nenhum argumento?

6.3.6 mkdir (make directory)

Cria novos diretórios (vazios).

Sintaxe básica:

```
$ mkdir [caminho1/diretório1] [caminho2/diretório2] ...
```

Exemplos

1. Para criar os diretórios “Pasta1” e “Pasta2” dentro do diretório /tmp, fazemos:

```
$ mkdir /tmp/Pasta1 /tmp/Pasta2
```

Naturalmente, se estivéssemos dentro do diretório /tmp, não seria necessário usar o caminho absoluto:

```
$ pwd  
/tmp  
$ mkdir Pasta1 Pasta2
```

6.3.7 rmdir (remove directory)

Remove um ou mais diretórios vazios.

Sintaxe básica:

```
$ rmdir [caminho1/diretório1] [caminho2/diretório2] ...
```

Exemplos

1. Para remover os diretórios “Pasta1” e “Pasta2” criados como nos exemplos do comando `mkdir`, poderíamos usar:

```
$ rmdir /tmp/Pasta1 /tmp/Pasta2
```

Exercícios

1. Vá até seu diretório home e crie um diretório chamado “Teste”. Use o comando `ls` para ver que o diretório foi criado. Remova o diretório criado e use novamente o comando `ls` para ver que a pasta foi removida.

6.3.8 touch

Pode ser usado para criar novos arquivos vazios e também para mudar a data e a hora de criação de arquivos existentes.

Sintaxe básica:

```
touch [opções] [arquivo1] [arquivo2] ...
```

Exemplos

1. Para criar um arquivo vazio chamado “arquivonovo” no diretório atual, poderíamos usar:

```
$ touch arquivonovo
```

Opções

- **-t** `[[YY]YY]MMDDhhmm[.ss]` - Altera a data e hora do arquivo para o ano YYYY (nesse caso, pode-se usar os quatro dígitos ou apenas dois), para o mês MM, para o dia DD, para a hora hh, para o minuto mm e para o segundo ss. Lembrando que as opções de ano e segundo são opcionais (por isso foram colocadas entre colchetes).

Exemplos

1. Para alterar a data do arquivo “arquivonovo” para o dia 16/11 (16 de novembro), e o horário para 16h11min, usamos:

```
$ touch -t 11161611 arquivonovo
```

2. Suponhamos que quiséssemos alterar os segundos também (para 11, por exemplo):

```
$ touch -t 11161611.11 arquivonovo
```

3. Por fim, se quiséssemos que a data do arquivo “arquivonovo” fosse 01/01/2013, com horário 0h0min, rodaríamos o comando da seguinte forma:

```
$ touch -t 201301010000 arquivonovo
```

Exercícios

1. Qual é a diferença entre o resultado produzido pelos dois comandos a seguir?

```
$ touch "arquivo novo"
```

e

```
$ touch arquivo novo
```

2. Crie um arquivo chamado “teste” em seu diretório home, usando o comando touch. Use ls (com a opção -l) para ver a data do novo arquivo criado. Mude a data e o horário do arquivo para o seu nascimento e use o comando ls para ver a nova data do arquivo.

6.3.9 rm (remove)

Remove arquivos e diretórios.

Sintaxe básica:

```
$ rm [opções] [arquivo1] [arquivo2] ...
```

Exemplos:

1. Criamos um arquivo chamado “teste” no diretório /tmp:

```
$ touch /tmp/teste
```

Agora vamos removê-lo:

```
$ rm /tmp/teste
```

Opções

- **-r**: Opção usada para remover recursivamente diretórios e seu conteúdo. Pode ser usada também para remover diretórios vazios.

Exemplos

1. Vamos criar um diretório (vazio) chamado “Pasta”.

```
$ mkdir Pasta
```

Se usarmos o seguinte comando para removê-lo, veremos um erro e o diretório não será removido:

```
$ rm Pasta  
ERRO!
```

Para removê-lo, teríamos que fazer:

```
$ rm -r Pasta
```

Poderíamos também usar o comando rmdir já apresentado.

2. Vamos criar agora um diretório chamado “Pastateste” dentro do diretório /tmp. E dentro de “Pastateste”, isto é, no /tmp/Pastateste, vamos criar um arquivo chamado “Arquivoteste”. Depois vamos removê-los.

```
$ mkdir /tmp/Pastateste
$ touch /tmp/Pastateste/Arquivoteste
```

Para remover, poderíamos fazer da seguinte maneira:

```
$ rm /tmp/Pastateste/Arquivoteste
$ rmdir /tmp/Pastateste
```

Mas a opção -r do comando rm nos permite remover o diretório e todo o seu conteúdo. Por isso, o comando a seguir já seria suficiente para remover o diretório “Pastateste” e seu conteúdo (no caso, o arquivo “Arquivoteste”).

```
$ rm -r /tmp/Pastateste
```

Atenção: O comando rm é definitivo, ou seja, uma vez que o usuário removeu um arquivo (ou um diretório), este não poderá ser recuperado. Não funciona simplesmente como uma lixeira, mas sim remove definitivamente o que for passado como argumento.

6.3.10 cp (copy)

Este comando serve para copiar arquivos.

Sintaxe básica:

```
$ cp [opções] [origem] [destino]
```

Exemplos

1. Para copiar o arquivo “teste” do /tmp para o diretório home do usuário:

```
$ cp /tmp/teste ~
```

Opções

- **-R:** Copia recursivamente os subdiretórios e seu conteúdo.

Exemplos

1. Suponha que um usuário possui um diretório no /tmp (/tmp/diretorio) e quer copiá-lo para sua home. Suponha ainda que esse diretório a ser copiado não está vazio.

```
$ cd /tmp/diretorio
$ ls
arquivo
$ cp -R /tmp/diretorio ~
```

6.3.11 mv (move)

Move e renomeia arquivos e diretórios.

Sintaxe básica

```
$ mv [opções] [origem] [destino]
```

Exemplos

1. Suponha que um usuário possui um arquivo em sua home chamado `arquivo1`. Para renomear este arquivo para `arquivonovo`, supondo que o usuário está em sua home, bastaria usar:

```
$ mv arquivo1 arquivonovo
```

2. Suponhamos agora que queremos mover o “`arquivonovo`” para o diretório `/tmp`. Para isso, o seguinte comando seria eficaz (estamos supondo ainda que o usuário está em sua home):

```
$ mv arquivonovo /tmp/
```

Após a execução desse comando, “`arquivonovo`” estaria no diretório `/tmp` e não haveria mais uma cópia do arquivo no diretório home do usuário.

Opções

- **-r**: Como outros comandos, essa opção move diretórios e seu conteúdo recursivamente.

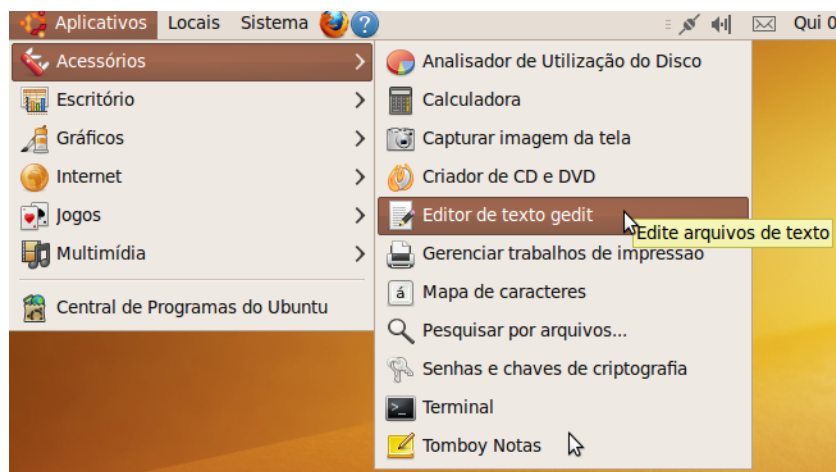
6.3.12 cat (concatenate)

Concatena arquivos e imprime o resultado no terminal.

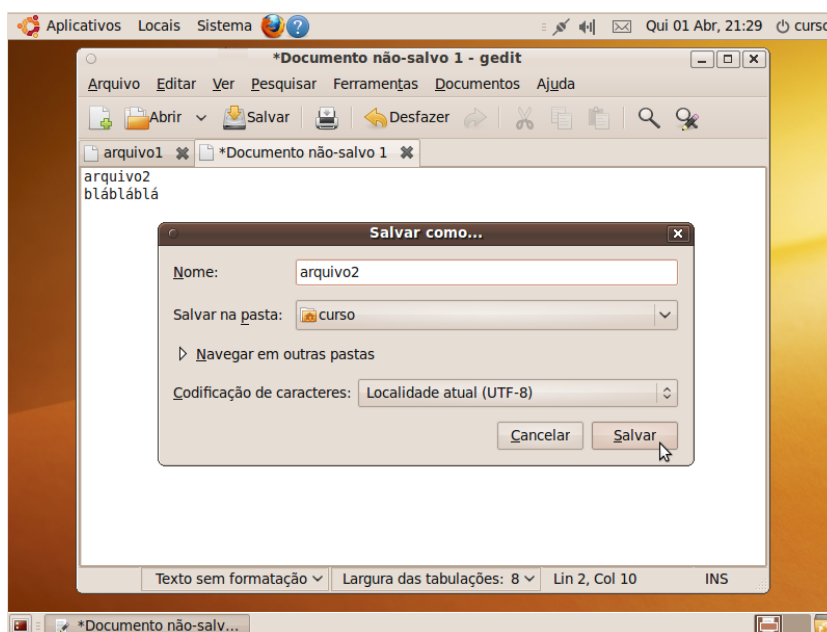
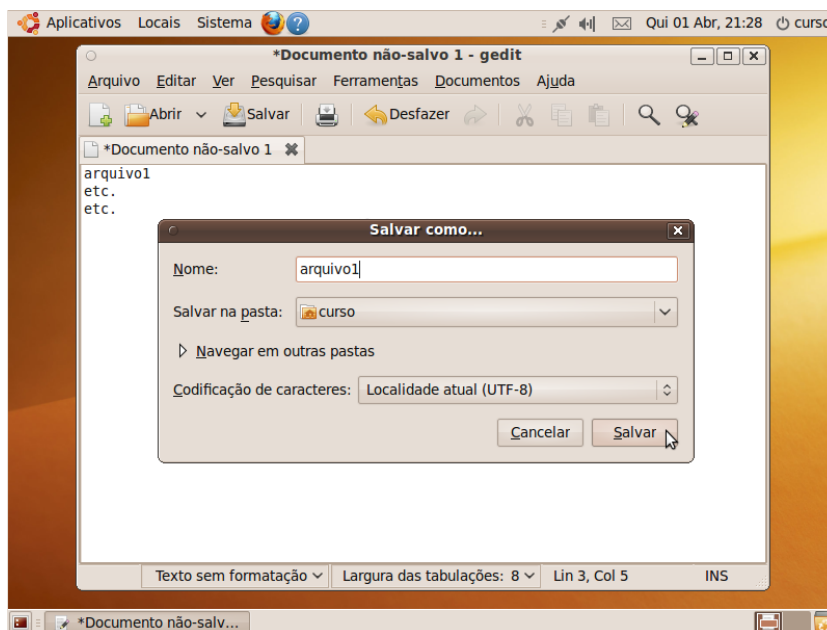
Sintaxe básica

```
$ cat [arquivo1] [arquivo2] ...
```

Para ilustrar o uso deste comando, vamos primeiro criar dois arquivos de texto não-vazios. Para isso, abra um editor de texto - pode ser qualquer um, utilizaremos o `gedit` por ser bastante simples.



Crie dois arquivos (`arquivo1` e `arquivo2`), contendo qualquer texto e salve-os no diretório home do usuário.



Observe agora o uso do comando cat:

```

curso@curso-desktop:~$ pwd
/home/curso
curso@curso-desktop:~$ ls
Área de Trabalho  arquivo2      Downloads      exemplo  Modelos  Público
arquivo1          Documentos    examples.desktop  Imagens  Música   Vídeos
curso@curso-desktop:~$ cat arquivo1
arquivo1
etc.
etc.
curso@curso-desktop:~$ cat arquivo2
arquivo2
blábláblá
curso@curso-desktop:~$ cat arquivo1 arquivo2
arquivo1
etc.
etc.

```

```
arquivo2
blábláblá
curso@curso-desktop:~$ cat arquivo2 arquivo1
arquivo2
blábláblá
arquivo1
etc.
etc.
```

6.3.13 find

O comando `find` é usado para procurar por diretórios e arquivos no disco. Possui várias opções, mas mostraremos apenas alguns exemplos simples.

Exemplos

1. Este exemplo procura por um arquivo ou diretório com o nome “Documents” a partir do / (diretório root):

```
$ find / -name Documents
```

2. Este outro procura por um arquivo ou diretório com o nome “Music” a partir do diretório home do usuário:

```
$ find ~ -name Music
```

É importante salientar que “a partir do diretório x” significa que o comando procurará dentro tudo o que estiver contido no tal diretório, incluindo os arquivos e os subdiretórios, bem como seu conteúdo e assim por diante.

6.3.14 clear

Use o comando `clear` e descubra o que ele faz:

```
$ clear
```

6.3.15 exit

Este comando serve para sair do shell (interpretador) e para efetuar o log out do usuário no terminal.

6.3.16 echo

Mostra um texto. Por agora, pode parecer um comando pouco útil, mas é bastante usado sobretudo em scripts para exibir mensagens ao usuário.

Sintaxe básica

```
$ echo mensagem
```

Exemplos

1. Note que a primeira linha corresponde ao comando, a segunda, ao resultado da execução deste comando:

```
$ echo mensagem
mensagem
```

2. Mais um exemplo:

```
$ echo Uma mensagem mais comprida
Uma mensagem mais comprida
```

Exercícios

1. Rode o comando a seguir e tenha certeza de que entendeu sua saída.

```
$ echo ~
```

2. Rode os comandos a seguir e observe a diferença entre seus resultados.

```
$ echo "aspas"

$ echo \"aspas\"
```

6.3.17 date

O comando `date` imprime ou modifica a data e o horário do sistema. É importante salientar que somente o usuário `root` e usuários privilegiados podem rodar este comando.

Sintaxe básica:

```
$ date [data]
```

Exemplos

1. Para visualizar a data e a hora do sistema:

```
$ date
Mon Mar  8 14:45:21 BRT 2010
```

2. Para alterar a data e a hora do sistema, basta usar o comando da seguinte maneira:

```
$ date MMDDhhmm[[YYyy][.ss]]
```

Onde `MM` é o mês, `DD` é o dia, `hh` é a hora, `mm` são os minutos. Opcionalmente, podem ser usados o ano (com 2 ou 4 dígitos) e os segundos (`ss`). Para alterar a data do sistema para o dia 1 de fevereiro e o horário para 14:30, poderíamos fazer:

```
$ date 02011430
```

6.3.18 chmod (change mode)

Este comando é usado para mudar permissões de arquivos ou diretórios.

Sintaxe básica:

```
$ chmod [permissões] [diretório/arquivo]
```

Há duas formas de usar o comando.

A primeira forma é bem simples. Você precisa saber que “u” representa o dono (“user”), “g”, o grupo, “o”, os demais usuários e “a”, por sua vez, representa todos (“all”). As letras “r”, “w” e “x” são as permissões apresentadas anteriormente. Além disso, você precisa saber que “+” acrescenta uma permissão, ao passo que “-” retira. Se usarmos “=”, teremos uma permissão exata. Vamos examinar alguns exemplos para podermos entender melhor.

Exemplos

Consideremos o arquivo exemplo (aquele que apareceu no comando ls), cuja permissão era rw-r--r--. Consideremos ainda que estamos no diretório home do usuário curso (/home/curso).

1. Suponhamos que queremos acrescentar permissão de escrita ao grupo. Poderíamos fazer isso da seguinte forma:

```
$ chmod g+w exemplo
```

2. Suponhamos agora que acabamos de nos arrepender e queremos tirar a permissão de escrita para o grupo. Poderíamos fazer da seguinte forma:

```
$ chmod g-w exemplo
```

3. Para acrescentar a permissão de execução a todos os usuários, fazemos:

```
$ chmod a+x exemplo
```

4. Para que os demais usuários fiquem sem permissão de leitura, mas tenham permissão de escrita e execução, temos:

```
$ chmod o=wx exemplo
```

O outro modo de alterar permissões é usando o chamado modo octal. Para usá-lo, é preciso ter em mente o seguinte:

- 0 - Nenhuma permissão de acesso.
- 1 - Permissão de execução.
- 2 - Permissão de escrita.
- 4 - Permissão de leitura.

A partir disso, podemos obter qualquer permissão, somando os números correspondentes às permissões desejadas.

- 3 - Permissão de execução e escrita (1 + 2).
- 5 - Permissão de execução e leitura (1 + 4).
- 6 - Permissão de escrita e leitura (2 + 4).
- 7 - Todas as permissões: execução, escrita e leitura (1 + 2 + 4).

Com esses algarismos, construímos números com três dígitos (XYZ, onde X representa a permissão que será definida para o dono, Y, a permissão do grupo, e Z é a permissão para outros usuários). Vamos mostrar como usar o modo octal.

Exemplos

1. Observe o exemplo a seguir:

```
$ chmod 762 exemplo
```

ou

```
$ chmod 762 /home/curso/exemplo
```

Nesse caso, estamos dando permissão 7 ao dono do arquivo exemplo, isso significa que estamos dando permissão de leitura, escrita e execução ao dono do arquivo. Para o grupo, demos permissão 6 (escrita e leitura). Aos demais, demos apenas permissão de escrita (permissão 2).

Vale lembrar que este comando (como outros) aceita caminhos relativos e absolutos.

Exercícios

1. Como você daria permissão de escrita e leitura para o dono do arquivo exemplo, permissão de leitura para o grupo e nenhuma permissão para os demais usuários, usando o modo octal?
2. Como você daria permissão de leitura e escrita a todos os usuários usando o primeiro modo apresentado?

6.3.19 passwd (password)

Este comando é usado para atualizar informações dos usuários.

Neste módulo, diremos apenas que ele pode ser usado para alterar a senha do seu próprio usuário.

Sintaxe básica

```
$ passwd
```

Após digitar este comando no terminal, o usuário deverá digitar sua senha atual (lembrando que não haverá nenhuma evidência - como asteriscos ou pontos - de que o usuário está digitando), depois a nova senha e, por último, será pedido para que o usuário confirme a nova senha.

6.3.20 su

O comando su é usado para mudar de usuário ou para tornar-se superuser (administrador do sistema ou usuário root).

Sintaxe básica

```
$ su [usuário]
```

Exemplos

1. Suponha que você esteja “logado” num terminal como “usuarioa” e deseja logar-se como “usuariob”, sem ter que encerrar a sessão como “usuarioa”:

```

$ whoami
usuarioa
$ su usuariob
Senha:
$ whoami
usuariob
$ exit
exit
$ whoami
usuarioa

```

Acompanhe a sequência de comandos: a princípio, o usuário que estava logado era “usuarioa”, o que pôde ser confirmado pelo comando `whoami`. A seguir, para mudar sua identidade para “usuariob”, o comando `su` foi utilizado - note que foi preciso digitar a senha de usuariob. Depois de autenticado, o usuário logado passou a ser “usuariob”. Com o comando `exit` fechou-se a sessão de “usuariob” e a identidade voltou a ser “usuarioa”.

2. Para tornar-se o usuário `root`, basta usar o comando `su` sem nenhum argumento:

```

$ su
Senha:
#

```

Note que foi necessário digitar a senha do usuário `root`.

6.3.21 sudo (su “do”)

Comando usado para obter privilégios de outros usuários (sobretudo do usuário `root`) para executar determinadas tarefas.

Algumas tarefas como instalar programas, alterar configurações essenciais do sistema etc. não podem ser desempenhadas por qualquer usuário, mas apenas pelo usuário `root` e/ou por alguns outros usuários que possam utilizar o comando `sudo` (os chamados `sudoers`).

No Ubuntu 9.10, o usuário criado no momento da instalação é um `sudoer` e não é criada uma senha para usuário `root`. Isso significa que, para desempenhar tarefas administrativas é necessário acrescentar “`sudo`” à frente do comando.

Observe o seguinte exemplo:

```

$ whoami
curso
$ shutdown -h now
shutdown: Precisa ser root
$ sudo shutdown -h now
[sudo] password for curso:

```

O usuário `curso` gostaria de desligar seu computador através da linha de comando, usando o comando `shutdown`. Acontece que, para executar tal comando, é necessário ser `root`. Por ser um `sudoer`, o usuário `curso` utilizou o comando `sudo` (observe que foi preciso digitar a senha do usuário `curso`) e conseguiu desligar o computador.

6.3.22 wc

O comando `wc` é usado para contar linhas, palavras e bytes de um arquivo ou do que for escrito no terminal.

Sintaxe básica

```
$ wc [opções] [arquivo]
```

Opções

- **-c:** Imprimir a contagem de bytes.
- **-l:** Imprimir o número de linhas.
- **-w:** Imprimir o número de palavras.

Exemplos

Vamos usar, para estes exemplos, o conteúdo dos arquivos “arquivo1” e “arquivo2”, mostrados na explicação do comando cat.

1. Para exibir o número de linhas do arquivo “arquivo1”, usaríamos:

```
$ wc -l arquivo1
3 arquivo1
```

2. Para exibir o número de palavras e de bytes do arquivo “arquivo2”:

```
$ wc -wc arquivo2
2 22 arquivo2
```

3. Se usássemos o comando wc sem nenhuma opção para “arquivo1”, obteríamos:

```
$ wc arquivo1
3 3 19 arquivo1
```

onde o primeiro número é a contagem de linhas, o segundo, de palavras, e o terceiro, o de bytes.

6.4 Pipe e redirecionamento

Além dos comandos apresentados anteriormente, a linha de comando ainda possui outros recursos para facilitar tarefas. Nesta seção, apresentaremos algumas ferramentas de direcionamento de entrada e saída.

6.4.1 | (Pipe)

O pipe (|) é usado para fazer encadeamento de processos, ou seja, faz com que a saída de um comando seja enviada como entrada para o próximo comando.

Observe o exemplo a seguir para entender melhor (o conteúdo de “arquivo1” e “arquivo2” é aquele que foi apresentado junto com o comando cat):

```
$ cat arquivo1 arquivo2 | wc -l
5
```

Vamos esclarecer o que aconteceu na execução deste comando: primeiro, utilizamos o comando cat com dois arquivos como argumento. Se rodássemos apenas este comando, teríamos o seguinte efeito (lembra-se?):

```
$ cat arquivo1 arquivo2
arquivo1
etc.
etc.
arquivo2
blábláblá
```

Mas acrescentamos um pipe (|) após a execução deste comando, o que significa que a saída foi redirecionada para o próximo comando, isto é, o resultado da execução de “cat arquivo1 arquivo2” não foi impressa, mas sim serviu como entrada para o próximo comando, “wc -l” - que contou o número de linhas e imprimiu este resultado no terminal.

Vamos mostrar agora um exemplo mais interessante:

```
curso@curso-desktop:~$ ls -l | wc -l
13
```

O comando antes do pipe lista o conteúdo do diretório atual, exibindo um item por linha. Se executássemos apenas este comando, obteríamos o seguinte resultado:

```
curso@curso-desktop:~$ ls -l
Área de Trabalho
arquivo1
arquivo2
doc
Documentos
Downloads
examples.desktop
exemplo
Imagens
Modelos
Música
Público
Vídeos
```

Mas em vez desta saída ser impressa, ela foi direcionada ao comando “wc -l”, que contou o número de linhas. Em outras palavras, o que o comando “ls -l | wc -l” fez foi contar o número de arquivos e diretórios dentro do diretório atual.

6.4.2 >

Esta é uma outra forma de direcionar a saída de um comando: diferente do |, que direcionava a saída de um comando para um outro programa ou comando, o > direciona a saída de um comando para um arquivo ou dispositivo.

Exemplos

1. O comando a seguir redireciona a saída de “cat arquivo1” para um arquivo chamado “arquivo3”:

```
curso@curso-desktop:~$ ls
Área de Trabalho  doc          examples.desktop  Modelos  Vídeos
arquivo1          Documentos   exemplo           Música
arquivo2          Downloads   Imagens           Público
curso@curso-desktop:~$ cat arquivo1
arquivo1
```



```

etc.
etc.
curso@curso-desktop:~$ cat arquivo1 > arquivo3
curso@curso-desktop:~$ ls
Área de Trabalho  arquivo3      Downloads      Imagens  Público
arquivo1          doc           examples.desktop Modelos  Vídeos
arquivo2          Documentos    exemplo        Música
curso@curso-desktop:~$ cat arquivo3
arquivo1
etc.
etc.

```

Observe que o arquivo “arquivo3” não existia, foi criado quando da execução do comando “cat arquivo1 > arquivo3”. Se o arquivo “arquivo3” já existisse, seu conteúdo seria sobrescrito.

2. Observe agora que “arquivo3” já existe:

```

curso@curso-desktop:~$ ls
Área de Trabalho  arquivo3      Downloads      Imagens  Público
arquivo1          doc           examples.desktop Modelos  Vídeos
arquivo2          Documentos    exemplo        Música
curso@curso-desktop:~$ cat arquivo3
arquivo1
etc.
etc.
curso@curso-desktop:~$ cat arquivo2 > arquivo3
curso@curso-desktop:~$ ls
Área de Trabalho  arquivo3      Downloads      Imagens  Público
arquivo1          doc           examples.desktop Modelos  Vídeos
arquivo2          Documentos    exemplo        Música
curso@curso-desktop:~$ cat arquivo3
arquivo2
blábláblá

```

6.4.3 >>

O >>, assim como o >, também direciona a saída de um comando para um arquivo, a diferença é que ele não substitui o conteúdo do arquivo, mas acrescenta ao final.

```

curso@curso-desktop:~$ ls
Área de Trabalho  arquivo3      Downloads      Imagens  Público
arquivo1          doc           examples.desktop Modelos  Vídeos
arquivo2          Documentos    exemplo        Música
curso@curso-desktop:~$ cat arquivo3
arquivo2
blábláblá
curso@curso-desktop:~$ cat arquivo1 >> arquivo3
curso@curso-desktop:~$ cat arquivo3
arquivo2
blábláblá
arquivo1
etc.
etc.

```

Exercício: Pesquise sobre outras formas de redirecionamento: < e <<.

6.5 Instalando programas pela linha de comando

Já mostramos como instalar programas usando o Synaptic, agora mostraremos como fazer isso através da linha de comando. Para isso, utilizaremos uma ferramenta chamada apt-get.

Tanto o Synaptic quanto o apt-get são baseados no APT (Advanced Packaging Tool), que é um gerenciador de pacotes que permite instalar e atualizar programas de forma prática, resolvendo dependências automaticamente. Convém salientar que o APT (assim como o apt-get e o Synaptic) está presente em várias distros, como Debian e Ubuntu.

Com o apt-get é possível, portanto, instalar, remover e atualizar programas.

Para usar o apt-get, o primeiro passo é rodar o comando “apt-get update”, que faz com que o apt-get baixe a lista com os pacotes disponíveis. Isso permite que ele crie uma espécie de banco de dados com os pacotes disponíveis, onde cada um pode ser encontrado e qual endereço contém a versão mais recente. Este comando deve ser executado periodicamente. O ideal é que você o use uma vez por semana, ou sempre que for fazer alguma instalação importante:

```
# apt-get update
```

Note que foi preciso executar tal comando como root. Você também poderia executá-lo usando sudo:

```
$ sudo apt-get update
```

Depois disso, você poderá instalar os programas desejados, usando a seguinte sintaxe:

```
# apt-get install [nome do programa]
```

ou

```
$ sudo apt-get install [nome do programa]
```

Para desinstalar um programa, também é muito simples:

```
# apt-get remove [nome do programa]
```

ou

```
$ sudo apt-get remove [nome do programa]
```

Finalmente, existe a opção de atualizar todo o sistema, o que é feito usando os comandos:

```
# apt-get update  
# apt-get upgrade
```

O “apt-get update” é o comando que baixa a lista dos pacotes disponíveis, como já vimos. O “apt-get upgrade”, por sua vez, age de forma bem diferente: ele verifica todos os pacotes do sistema e tenta atualizar todos de uma vez, o que geralmente resulta em uma longa lista de atualizações.

Capítulo 7

Obtendo ajuda

O que foi apresentado neste curso tem caráter introdutório: mostramos neste capítulo algumas formas de se aprofundar e de achar respostas para alguns problemas.

7.1 Comandos e opções

Através da própria linha de comando é possível obter ajuda e informações a respeito dos comandos.

7.1.1 man (manual)

O comando `man` mostra uma página de manual para um determinado comando.

Sintaxe básica

```
$ man [comando]
```

Para visualizar o manual do comando `ls`, basta usar

```
$ man ls
```

Para sair de uma página de manual, basta digitar “q”.

7.1.2 apropos

Este comando faz buscas de palavras em um banco de dados que contém descrições curtas de comandos e programas.

Sintaxe básica

```
$ apropos [busca]
```

Suponhamos que quiséssemos procurar como remover arquivos. Poderíamos usar

```
$ apropos remove
```

Provavelmente, esta busca retornaria muitos resultados. Sejamos então mais específicos:

```
$ apropos "remove files"
```

Esta busca retornaria o seguinte resultado:

```
rm                (1) - remove files or directories
```

7.1.3 --help

Quase todos os comandos do GNU/Linux possuem a opção “--help”, usada, obviamente, para obter ajuda sobre o comando em questão.

Sintaxe básica

```
$ [comando] --help
```

Para obtermos ajuda sobre o `wc`, por exemplo, usamos

```
$ wc --help
```

7.2 Internet e literatura

É até dispensável mencionar que você sempre pode pesquisar na internet quando encontrar algum problema ou dúvida em relação ao GNU/Linux: existem muitos fóruns de discussões e tutorias, assim como a documentação dos programas, bibliotecas etc. também está disponível na rede.

Outra boa fonte de informações é a literatura: existem vários livros disponíveis sobre diversos tópicos do GNU/Linux. Sugerimos aqui os da editora O’Reilly (<http://oreilly.com/pub/topic/linux>).

7.3 Sugestões

Finalizamos este módulo com sugestões de sites que podem ajudá-lo a entender melhor o GNU/Linux.

- **Rede GNU/Linux:** Site da Rede GNU/Linux do Instituto de Matemática e Estatística da USP.
www.linux.ime.usp.br
- **Make The Move:** Tem como objetivo apresentar o Linux e o Software Livre como alternativas viáveis ao sistema em seu computador.
makethemove.net
- **Free Software Foundation:** Site da FSF.
www.fsf.org
- **Debian - The Universal Operating System:** Site da distro Debian.
www.debian.org
- **Ubuntu-BR:** Site da Comunidade Ubuntu brasileira.
www.ubuntu-br.org
- **Fedora Project:** Site da distro Fedora.
fedoraproject.org
- **Distro Watch:** Notícias e informações sobre distribuições Linux e BSD.
distrowatch.com
- **Google:** Dispensa apresentações.
www.google.com

Referências Bibliográficas

- [1] *1234407730filesystem.gif*, disponível in <http://www.linuxplanet.com/graphics/screenshots/1234407730filesystem.gif> [Fevereiro de 2010]
- [2] *Ficheiro: Fluxbox.png* - *Wikipédia, a enciclopédia livre*, disponível in <http://pt.wikipedia.org/wiki/Ficheiro:Fluxbox.png> [Janeiro de 2010]
- [3] *Ficheiro: KDE 4.png* - *Wikipédia, a enciclopédia livre*, disponível in http://pt.wikipedia.org/wiki/Ficheiro:KDE_4.png [Janeiro de 2010]
- [4] *Ficheiro: Linus Torvalds.jpeg* - *Wikipédia, a enciclopédia livre*, disponível in http://pt.wikipedia.org/wiki/Ficheiro:Linus_Torvalds.jpeg [Janeiro de 2010]
- [5] *Ficheiro: Richard Matthew Stallman.jpeg* - *Wikipédia, a enciclopédia livre*, disponível in http://pt.wikipedia.org/wiki/Ficheiro:Richard_Matthew_Stallman.jpeg [Janeiro de 2010]
- [6] *Filesystem Hierarchy Standard* - *Wikipédia, a enciclopédia livre*, disponível in http://pt.wikipedia.org/wiki/Filesystem_Hierarchy_Standard [Fevereiro de 2010]
- [7] *GNU General Public License* - *Wikipédia, a enciclopédia livre*, disponível in http://pt.wikipedia.org/wiki/GNU_General_Public_License [Janeiro de 2010]
- [8] SILVA, Gleydson Mazzioli da, *Guia Foca GNU/Linux*, novembro de 2007, disponível in <http://www.guiafoca.org/download/iniciante/focalinux1-pdf.tar.gz> [Janeiro de 2010]
- [9] *Linux* - *Wikipédia, a enciclopédia livre*, disponível in <http://pt.wikipedia.org/wiki/Linux> [Janeiro de 2010]
- [10] *O que é Linux*, disponível in <http://www.vivaolinux.com.br/linux/> [Janeiro de 2010]
- [11] CAMPOS, Augusto, *O que é Linux*, Florianópolis, março de 2006, disponível in <http://br-linux.org/faq-linux> [Janeiro de 2010]
- [12] MORIMOTO, Carlos E., *Tutorial completo do apt-get*, abril de 2007, disponível in <http://www.guiadohardware.net/tutoriais/tutorial-completo-apt-get/> [Março de 2010]
- [13] *Ubuntu System Requirements* - *Community Ubuntu Documentation*, disponível in <https://help.ubuntu.com/community/Installation/SystemRequirements> [Janeiro de 2010]