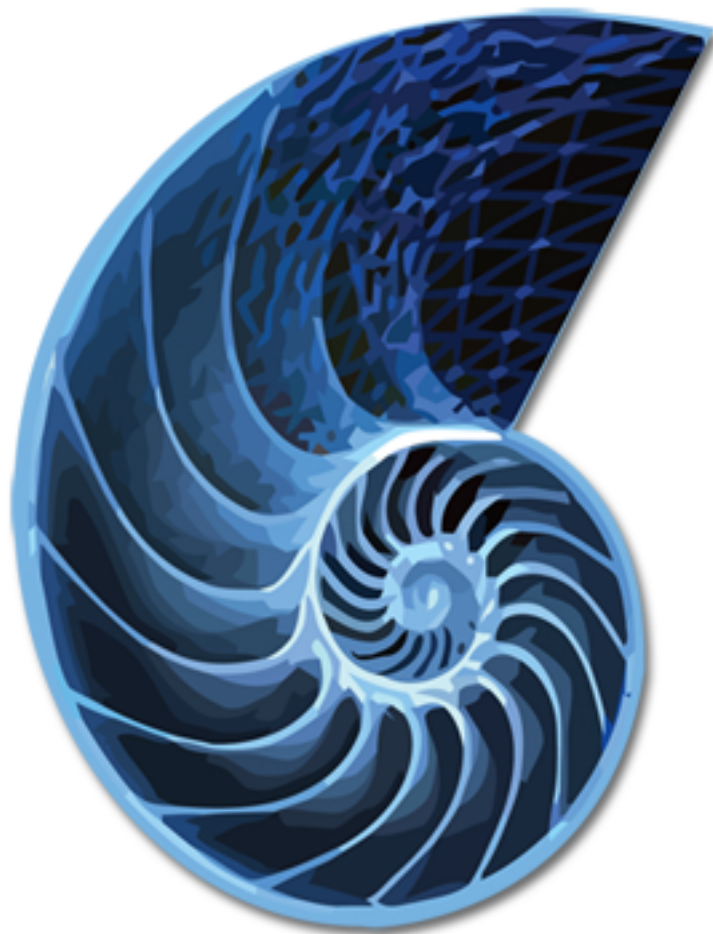


Linux Course

Shell Script



Eduardo Mondoni

Linux Course Shell Script

Edição 1

Autor

Eduardo Mondoni

mondoni@linustec.com.br

Copyright © 2012 Eduardo Mondoni.

O texto deste documento está licenciado por Linustec sob a licença Creative Commons Attribution–Share Alike 3.0 Unported ("CC-BY-SA"). Uma explicação de CC-BY-SA pode ser encontrada em <http://creativecommons.org/licenses/by-sa/3.0/>. The original authors of this document designate the Linux Course Project as the "Attribution Party" for purposes of CC-BY-SA. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Esta apostila tem como objetivo demonstrar a utilização do Bash para a criação de shell scripts. Esta apostila também será utilizada como referência para aprendizado do assunto durante os encontros do grupo de estudos **Linux Course**.

Preface	v
1. Convenções de Documentos	v
1.1. Convenções Tipográficas	v
1.2. Convenções de Pull-Quote	vi
1.3. Notas e Avisos	vii
2. Nós precisamos de seu feedback	viii
1. Introdução	1
1.1. Camadas do Linux	1
1.2. O que é o Shell?	1
1.3. Tipos de Shells	1
1.4. Vantagens do Bourne Again Shell (BASH)	2
1.5. Arquivos de inicialização do bash	2
1.5.1. Shell Interativo com login ou opção "--login"	2
1.5.2. Shell Interativo sem login	2
1.5.3. Shell não interativo	2
1.5.4. Execução com o comando sh	3
1.6. Executando comandos	3
1.6.1. Comandos built-in do Shell	3
1.6.2. Propriedades de bons scripts:	3
2. Criação de um Shell Script	5
2.1. Hello World	5
2.2. Variáveis	6
A. Referências	7
B. Revision History	9
Índice Remissivo	11

Preface

1. Convenções de Documentos

Este manual usa diversas convenções para destacar certas palavras e frases e chamar a atenção para informações específicas.

Em PDF e edições de texto, este manual usa tipos de letras retiradas do conjunto de Fontes [Liberation Fonts](https://fedorahosted.org/liberation-fonts/)¹. O conjunto de Fontes Liberation Fonts, também é usado em formato HTML, caso o conjunto esteja instalado em seu sistema. Caso ainda não esteja, como forma alternativa, estão disponíveis tipos de letras equivalentes. Nota: O Red Hat Enterprise Linux 5 e versões mais recentes do mesmo, incluem o conjunto Liberation Fonts por padrão.

1.1. Convenções Tipográficas

São usadas quatro convenções tipográficas para realçar palavras e frases específicas. Estas convenções, e circunstâncias a que se aplicam, são as seguintes:

Negrito Espaço Único (Mono-spaced Bold)

Usada para realçar entradas do sistema, incluindo comandos de shell, nomes de arquivos e caminhos. São também usadas para realçar tecla de Maiúsculas/Minúsculas e as combinações de teclas. Por exemplo:

Para ver o conteúdo do arquivo **my_next_bestselling_novel** em sua pasta de trabalho atual, insira o comando **cat my_next_bestselling_novel** na janela de solicitação e pressione **Enter** para executar o comando.

O comando acima inclui um nome de arquivo, um comando de shell e uma tecla **cap**, todos apresentados em **Negrito Espaço Único** (Mono-spaced Bold) e todos distintos, graças ao conteúdo.

As combinações de Teclas podem ser diferenciadas da tecla Caps Lock por um hífen, conectando cada parte de uma combinação de teclas. Por exemplo:

Pressione **Enter** para executar o comando.

Pressione **Ctrl+Alt+F2** para mudar para o primeiro terminal virtual. Pressione **Ctrl+Alt+F1** para voltar para sua sessão do X-Windows.

A primeira sentença, destaca uma tecla específica a ser pressionada. A segunda destaca duas combinações de teclas (cada conjunto de três teclas Caps com cada um pressionado simultaneamente).

Caso o código fonte seja discutido, serão apresentados como acima, os nomes de classe, métodos, funções, nomes de variantes e valores retornados mencionados em um parágrafo, em **Negrito de Espaço Único** (Mono-spaced Bold). Por exemplo:

Classes baseadas em arquivo, incluem **filesystem** para sistemas de arquivo, **file** para arquivos, e **dir** para diretórios. Cada classe possui seu conjunto próprio de permissões associadas.

¹ <https://fedorahosted.org/liberation-fonts/>

Negrito Proporcional

Esta representa as palavras e frases encontradas no sistema, incluindo os nomes de aplicativos, texto de caixa de diálogo, botões rotulados, caixa de seleção e rótulos de botão de opção, títulos de menus e sub-menus. Por exemplo:

Escolha **Sistema** → **Preferências** → **Mouse** a partir da barra de menu para obter **Preferências de Mouse**. Na aba **Botões**, clique na caixa de seleção **Mouse para mão esquerda** e clique em **Fechar** para mudar o botão do mouse anterior da esquerda para a direita (deixando-o assim, mais adequado para o uso do canhoto).

Para inserir um caractere especial em um arquivo **gedit**, escolha **Aplicativos** → **Acessórios** → **Mapa de Caracteres** a partir da barra de menu principal. Depois, escolha a opção **Pesquisar** → **Encontrar...** a partir da barra de menu **Mapa de Caracteres**, digite o nome do caractere no campo **Pesquisar** e clique em **Próximo**. O caractere que você buscou estará em destaque na **Tabela de Caracteres**. Clique novamente nestes caracteres realçados para colocá-los no campo **Texto para cópia** e depois clique em **Copiar**. Agora mude novamente para seu documento e escolha **Editar** → **Colar** a partir da barra de menu do **gedit**.

O texto acima inclui nomes de aplicativos, nomes de menu e itens de todo o sistema, nomes de menu específicos do aplicativo, e botões e textos encontrados na Interface Gráfica GUI, todos apresentados em Negrito Proporcional (Proportional Bold) e todos diferenciados de acordo com o contexto.

Itálico em Negrito de Espaço Único (Mono-spaced Bold Italic) ou ***Itálico em Negrito Proporcional (Proportional Bold Italic)***

Sendo o Negrito Espaço Único (Mono-spaced Bold) ou Negrito Proporcional (Proportional Bold), os itálicos extras indicam textos substituíveis ou variáveis. O Itálico denota o texto que você não inseriu literalmente ou textos exibidos que mudam dependendo das circunstâncias. Por exemplo:

Para conectar-se à uma máquina remota usando o ssh, digite **ssh nome do usuário@domain.name** na janela de comandos. Por exemplo, considere que a máquina remota seja **example.com** e seu nome de usuário nesta máquina seja john, digite **ssh john@example.com**.

O comando **mount -o remount file-system** remonta o sistema de arquivo nomeado. Por exemplo, para remontar o sistema de arquivo **/home**, o comando é **mount -o remount /home**.

Para ver a versão de um pacote instalado, use o comando **rpm -q package**. Ele retornará um resultado como este: **package-version-release**.

Observe as palavras em negrito e itálicas acima — nome de usuário, nome do domínio, sistema de arquivo, pacote, versão e lançamento. Cada palavra é um espaço reservado, tanto para o texto que você inseriu ao emitir um comando ou para o texto exibido pelo sistema.

Além de uso padrão para apresentar o título de um trabalho, os itálicos denotam a primeira vez que um termo novo e importante é usado. Por exemplo:

O **Publican** é um sistema de publicação do *DocBook*.

1.2. Convenções de Pull-Quote

Resultado de terminal e listagem de código fonte são definidos visualmente com base no contexto.

O resultado enviado à um terminal é configurado em **Romano de Espaço Único (Mono-spaced Roman)** e apresentado assim:

```
books      Desktop  documentation  drafts  mss    photos  stuff  svn
books_tests Desktop1  downloads      images  notes  scripts svgs
```

As listas de código fonte também são configuradas em **Romano de Espaço Único (Mono-spaced Roman)**, porém são apresentadas e realçadas como a seguir:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;


public class ExClient
{
    public static void main(String args[])
        throws Exception
    {
        InitialContext iniCtx = new InitialContext();
        Object          ref    = iniCtx.lookup("EchoBean");
        EchoHome        home   = (EchoHome) ref;
        Echo             echo   = home.create();

        System.out.println("Created Echo");


        System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
    }
}
```

1.3. Notas e Avisos

E por fim, usamos três estilos visuais para chamar a atenção para informações que possam passar despercebidas.

 **Nota**

Uma nota é uma dica ou símbolo, ou ainda uma opção alternativa para a tarefa em questão. Se você ignorar uma nota, provavelmente não resultará em más consequências, porém poderá deixar passar uma dica importante que tornará sua vida mais fácil.

 **Importante**

Caixas importantes detalham coisas que são geralmente fáceis de passarem despercebidas: mudanças de configuração que somente se aplicam à sessão atual, ou serviços que precisam ser reiniciados antes que uma atualização seja efetuada. Se você ignorar estas caixas importantes, não perderá dados, porém isto poderá causar irritação e frustração.



Aviso

Um Aviso não deve ser ignorado. Se você ignorar avisos, muito provavelmente perderá dados.

2. Nós precisamos de seu feedback

Se você encontrar erro ortográfico neste manual, ou se você tem uma sugestão para melhoramento deste manual, nós gostaríamos muito de ouvir sua opinião! Por favor envie um email para mondoni@linustec.com.br referenciando como assunto: **Linux Course - Shell Script**.

Se você tiver uma sugestão para a melhora deste documento, por favor tente ser o mais específico possível em sua descrição. Caso você tenha encontrado um erro, por favor inclua o número da seção e alguns detalhes a respeito do texto para a nossa melhor identificação.

Introdução

1.1. Camadas do Linux

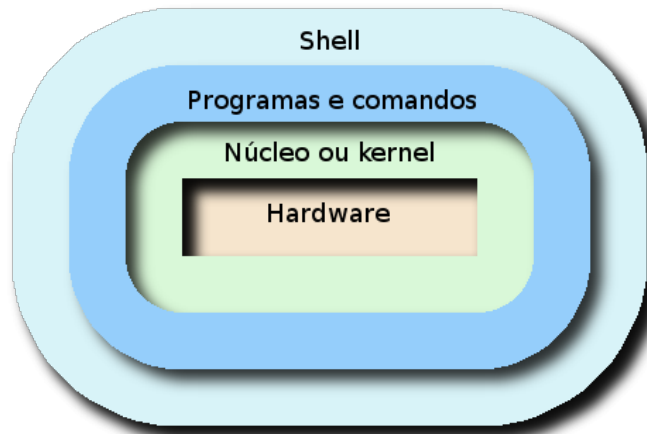


Figura 1.1. Camadas do Linux

Com base na [Figura 1.1, "Camadas do Linux"](#), podemos compreender quais as camadas que compõem um sistema operacional GNU/Linux desde o Hardware até o Shell.

- **Hardware:** composto pelos dispositivos físicos do sistema (teclado, mouse, monitor).
- **Kernel:** núcleo do sistema operacional sendo responsável pelo controle e gerenciamento do hardware.
- **Programas:** comandos que realizam tarefas bem específicas.
- **Shell:** responsável pela interação entre o usuário e o sistema operacional.

1.2. O que é o Shell?

Shell é o programa que reconhece o comando que você digitou e traduz para o sistema operacional qual a ação que será realizada, diminuindo o tempo de utilização do kernel para a execução da ação.

Por ser um interpretador de comandos de alto nível, o shell também pode ser utilizado como linguagem de programação. Onde o usuário pode combinar uma sequência de comandos criando um script para ser executar ao invés de sempre realizar a mesma sequência de comandos manualmente.

1.3. Tipos de Shells

- **Bourne Shell (sh):** Desenvolvido por Stephen Bourne da Bell Labs (da AT&T onde também foi desenvolvido o Unix), este foi durante muitos anos o Shell default do sistema operacional Unix. É também chamado de Standard Shell por ter sido durante vários anos o único e até hoje é o mais utilizado até porque ele foi portado para todos os ambientes Unix e distros Linux.
- **Korn Shell (ksh):** Desenvolvido por David Korn, também da Bell Labs, é um superset do sh, isto é, possui todas as facilidades do sh e a elas agregou muitas outras. A compatibilidade total com o sh vem trazendo muitos usuários e programadores de Shell para este ambiente.

- **Bourne Again Shell (bash)**: Este é o Shell mais moderno e cujo número de adeptos mais cresce em todo o mundo, seja por ser o Shell default do Linux, seu sistema operacional hospedeiro, seja por sua grande diversidade de comandos, que incorpora inclusive diversos instruções características do C Shell.
- **C Shell (csh)**: Desenvolvido por Bill Joy da Berkley University é o Shell mais utilizado em ambientes *BSD e Xenix. A estruturação de seus comandos é bem similar à da linguagem C. Seu grande pecado foi ignorar a compatibilidade com o sh, partindo por um caminho próprio. Além destes Shells existem outros, mas irei falar contigo somente sobre os três primeiros, tratando-os genericamente por Shell e assinalando as especificidades de cada um que porventura hajam.

1.4. Vantagens do Bourne Again Shell (BASH)

Bash é um shell compatível com sh que incorpora funções úteis de Korn shell (ksh) e C shell (csh). Ele está em conformidades com o padrão IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools. Oferece melhorias em relação ao sh tanto para programação quanto utilização interativa. Incluindo, edição de linha de comando, histórico de comandos ilimitado, controle de jobs entre outros. O bash pode executar a maioria dos scripts sh sem modificações.

1.5. Arquivos de inicialização do bash

Arquivos de inicialização são script que serão lidos e executados pelo Bash ao ser iniciado. A seguir uma breve explicação das diferentes maneiras de iniciar o bash e que arquivos serão lidos na inicialização.

1.5.1. Shell Interativo com login ou opção “--login”

Significa que você está logado num shell após ter efetuado login com usuário e senha, não necessariamente existirá um script sendo executado.

Arquivos lidos:

- `/etc/profile`
- `~/.bash_profile`, `~/.bash_login` ou `~/.profile`: será lido o primeiro arquivo existente.
- `~/.bash_logout`: lido no momento de efetua logout.

1.5.2. Shell Interativo sem login

Significa que você não precisou se autenticar no sistema. Exemplo, quando você abre um terminal a partir de um ícone ou um atalho de menu, isso é um shell sem login.

Arquivos lidos:

- `~/.bashrc`

Este arquivo é geralmente citado em `~/.bash_profile`:

```
if [ -f ~/.bashrc ]; then . ~/.bashrc; fi
```

1.5.3. Shell não interativo

Todos os script utilizando shell não interativo. Eles são programados para realizarem certas tarefas e não podem ser instruídos a realizarem outros trabalhos diferentes dos programados.

Arquivos lidos:

- definido por BASH_ENV.

1.5.4. Execução com o comando sh

O bash atua como o programa Bourne sh.

Arquivos lidos:

- /etc/profile
- ~/.profile

1.6. Executando comandos

O bash determina o tipo de programa que será executado. Programas normais são comandos do sistema que existem na forma compilada. Quando um desses programas é executado, um novo processo é criado porque o Bash faz uma cópia exata dele mesmo. Este procedimento chama-se forking.

1.6.1. Comandos built-in do Shell

São comandos existentes dentro do próprio shell. Quando um desses comandos é executado sendo a primeira palavra de uma linha de comando, o shell o executa diretamente sem a necessidade de criar um novo processo.

1.6.2. Propriedades de bons scripts:

1. Um script deve executar sem erros;
2. Deve executar a tarefa para qual foi criado;
3. Lógica de programação é claramente definida;
4. Um script não realiza tarefas desnecessárias;
5. Scripts devem ser reutilizáveis;

Criação de um Shell Script

Vamos então começar a criação dos nossos scripts utilizando comandos do nosso shell padrão, *Bash*, iniciando com elementos básicos de um shell script e ir complementando com comandos e exemplos práticos de nível intermediário e avançado. Conhecendo também os tipos de laços de repetição, controle de fluxos, redirecionamentos de saída de um comando para outro ou para um arquivo de log

2.1. Hello World

Como exemplo de nosso primeiro shell script simples iremos utilizando o **echo** para ecoar uma frase em nosso terminal.

```
$ echo 'Hello World!'
Hello World!
```

Seguindo o exemplo de utilização do comando **echo** acima, vamos transformá-lo em um shell script. Inicialmente devemos indicar no início do script qual será o interpretador de comandos que iremos utilizar. Como estamos utilizando o *bash* como shell padrão, iremos colocar "`#!/bin/bash`" na primeira linha do arquivo. O shell script de exemplo: [helloworld.sh](https://github.com/mondoni/linuxcourse/blob/master/shellscript/exemplos/helloworld.sh)¹.



Dica:

Evite utilizar acentos, espaços e caracteres especiais nos nomes de scripts.

```
$ vim helloworld.sh
#!/bin/bash
echo 'Hello World!'
```

Lembre-se de configurar a permissão de execução. E vamos executá-lo:

```
$ chmod +x helloworld.sh
$ ./helloworld.sh
Hello World!
```



Nota:

Seja qual for a linguagem a ser utilizada num script, sempre devemos indicar o interpretador de comandos adequadamente. Para um script em PHP utilizamos o interpretador "`/usr/bin/php`", para um script em Python utilizamos "`/usr/bin/python`", etc.

¹ <https://github.com/mondoni/linuxcourse/blob/master/shellscript/exemplos/helloworld.sh>

2.2. Variáveis

Para quem já conhece algum tipo de linguagem de programação já deve ter uma noção de como funciona esse processo. No *bash* não é muito diferente de outras linguagens de programação, aliás, ele é muito mais simples nesse quesito.

Para a atribuição de valores a uma variável o *bash* segue o padrão *variavel=valor* e para acessar a variável com seu conteúdo devemos utilizar o cifrão(\$) seguido do nome da variável *\$variavel*.

Vamos alterar nosso script "*helloworld.sh*" para que a mensagem esteja atribuída à uma variável.

```
$ vim helloworld.sh
#!/bin/bash
msg='Hello World!'
echo $msg

$ ./helloworld.sh
Hello World!
```

Apêndice A. Referências

Exemplos disponíveis em: <https://github.com/mondoni/linuxcourse/tree/master/shellscript/exemplos>

Neves, Julio Cezar. Programação Shell Linux, sétima edição, 2008.

Aurelio, Aurélio Marinho Jargas. Introdução a Shell Scripts. Disponível em: <http://aurelio.net/shell/apostila-introducao-shell.pdf>

Garrels, machtelt. Bash Beginners Guide, versão 1.1, 2008.

Abadi, Marcos. Uma Rapidinha nos Principais Sabores de Shell. 20/12/2011. Disponível em: <http://marcosabadi.blogspot.com/2011/12/uma-rapidinha-nos-principais-sabores-de.html>

Apêndice B. Revision History

Revisão 0-0 Fri Jan 6 2012

Eduardo Mondoni
mondoni@linustec.com.br

Initial creation of book by publican

Índice Remissivo

C

camadas

informações sobre camadas do linux., 1

F

feedback

contato de informação para este manual., viii

S

shell

definição do que é o shell., 1

