Programação Shell Script: como dominar seu terminal

Renê de Souza Pinto

14 de Outubro de 2008

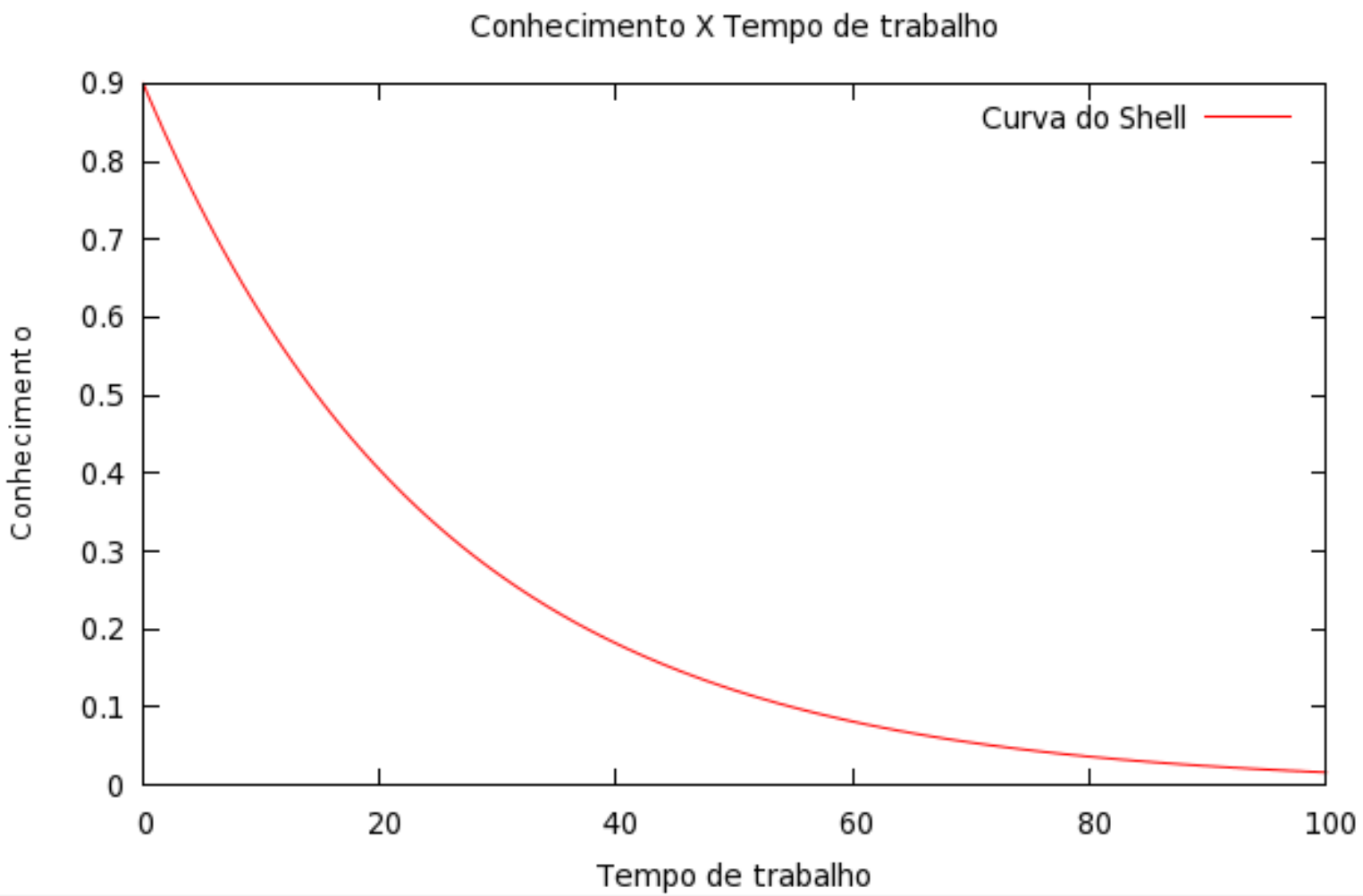
Motivação

- O que é Shell?
  - Programa interpretador de instruções
- Por que utilizar o Shell?

Motivação

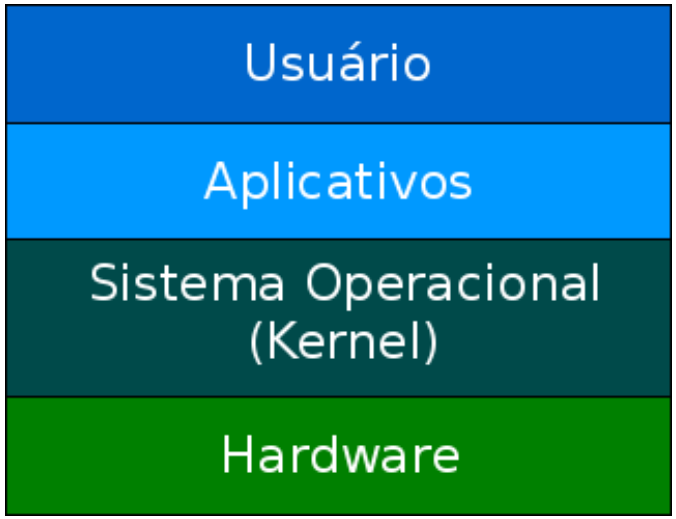
- Facilidade na **automatização** de tarefas
- Facilidade no tratamento de dados (inclusive em grandes quantidades)
- Rapidez no desenvolvimento
- Portabilidade em ambientes Unix
- Aplicações web com CGI
- Aplicações gráficas através do dialog, kdialog, etc...

Motivação


Conhecimento X Tempo de trabalho
Curva do Shell
Conhecimento
Tempo de trabalho
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
20
40
60
80
100

Introdução - Sistemas operacionais

- Divisão em camadas:



Introdução - Shell

- O Shell atua na camada de aplicativos
- É a interface entre o Kernel e o Usuário, ou seja, provê ao usuário as funcionalidades do Kernel através de um terminal extremamente robusto e poderoso
- Foi escritos em diferentes versões

Introdução - Shell

Versões de Shell:

- **Bourne Shell - sh**: É o Shell padrão do Unix, versão padrão escrita por Stephen Bourne da Bell Labs.
- **Bourne-Again Shell - bash**: Quase 100% compatível com o Bourne Shell, possui também algumas implementações feitas para o Korn Shell e comandos do C Shell.
- **Korn Shell - ksh**: Upgrade do Bourne Shell, escrito por David Korn, da Bell Labs.
- **C Shell - csh**: Possui uma sintaxe específica, não compatível com sh, bash, ksh.

Introdução - Shell

- Programar em Shell script implica em dominar os comandos do sistema

Comandos do terminal

- Programar em Shell script implica em dominar os comandos do sistema
- Vamos dominar nosso terminal!

Comandos do terminal - Obtendo ajuda

Comandos para pedir ajuda

help	Mostra informações gerais sobre os comandos internos (*built-ins*) do Shell.
man	Mais completa documentação do Linux
apropos	Mostra informações sobre um tópico
whatis	Uma breve descrição de um comando do sistema

Adaptado de [1]

Subdivisões das man-pages

1. Comandos de usuários	Comandos que podem ser executados a partir de um Shell
2. Chamadas de sistema	Chamadas implementadas pelo kernel
3. Bibliotecas de funções	A maioria das funções da biblioteca libc
4. Formatos de arquivos especiais	Drivers e hardware
5. Arquivos de configuração	Formatos de arquivos e convenções
6. Jogos e demonstrações	O próprio nome diz
7. Pacotes de macro e convenções	Sistemas de arquivos, protocolos de rede, códigos ASCII e outros.
8. Comandos de administração do sistema	Comandos que muitas vezes apenas o root pode executar.

Adaptado de [1]

Manipulação de arquivos

Comandos para manipulação de arquivos:

- **pwd** - Informa o diretório corrente
- **cd** - Troca de diretório
- **ls** - Lista arquivos
- **cp** - Copia arquivos
- **mv** - Move arquivos e diretórios
- **ln** - Cria links entre arquivos

Manipulação de arquivos

Comandos para manipulação de arquivos:

- **mkdir** - Cria diretórios
- **rmdir** - Remove um diretório **vazio**
- **rm** - Remove arquivos e diretórios
- **file** - Retorna o tipo de um arquivo
- **grep** - Busca conteúdo em arquivos
- **find** - Procura arquivos
- **basename** - Retorna o nome de um arquivo a partir de seu caminho completo
- **dirname** - Retorna o nome de um diretório recebendo seu caminho completo

Utilizando o find

O find é um poderoso comando para buscarmos arquivos

- Sintaxe: *find <caminho> [expressão] [ação]*
- expressão: Define os critérios de pesquisa:
  - -name <nome>: Procura arquivos com o nome <nome> (pode ser utilizado metacaracteres ou caracteres curingas)
  - -user <usuario>: Procura arquivos pertencentes a <usuario>
  - -group <grupo>: Procura arquivos pertencentes ao grupo <grupo>
  - -type c: Procura arquivos que tenham o tipo c, que pode ser:

Utilizando o find

- b - Arquivo especial de bloco
- c - Arquivo especial de caracter
- d - Diretório
- p - Named pipe (FIFO)
- f - Arquivo normal
- l - Link simbólico
- s - Socket

Utilizando o find

- -size ±d: Procura arquivos que usam mais de d unidades (+d) ou menos de d unidades (-d) de espaço. Aonde d pode ser:
  - b - Bloco de 512 bytes (default)
  - c - Caracteres
  - k - Kilobytes (1024 bytes)
  - w - Palavras (2 bytes)
- -atime ±d:
- -ctime ±d:
- -mtime ±d:

Funcionam com o mesmo princípio: Procuram por arquivos que foram acessados, cujo status mudou ou cujo os dados foram modificados há mais de d dias (+d) ou em menos de d dias (-d).

Utilizando o find

- -exec <comando> {};
  Executa uma ação para cada arquivo/diretório encontrado, {} é substituído pelo nome do arquivo.
  Ex: find /usr -type f -exec md5sum {};
  Calcula o *MD5* (Message-Digest algorithm 5) para cada arquivo de /usr

Utilizando o find

- Também é possível agrupar critérios de pesquisa:
  exp1 -**o** exp2
  ou
  exp1 -**a** exp2
  Aonde -**a** corresponde a um **AND** e -**o** a um **OR**.
- Ex: find /usr -type f -o -ctime +10

Exercícios

1. Exiba na tela informações (tipo de arquivo) do arquivo /bin/bash
2. Liste os arquivos da pasta /usr que foram modificados a menos de 2 dias
3. Liste os arquivos e diretórios em /home que foram modificados a menos de 1 dia ou cujo status mudou a mais de 10 dias.
4. Crie o diretório curso contendo um outro diretório (chamado shell) com apenas uma chamada do comando mkdir
5. Descubra o que o comando touch faz (Dica: consulte o man)
6. Crie o arquivo -**teste** (não esqueça do hífen) no diretório shell (criado anteriormente)
7. Remova o arquivo criado no exercício anterior

Respostas

1. file /bin/bash
2. find /usr/ -type f -mtime -2
3. find /home -mtime -1 -o -ctime +10
4. mkdir -p curso/shell
5. touch: Altera o tempo de acesso/modificação de um arquivo. É utilizado normalmente para se criar um arquivo vazio
6. touch – -teste
7. rm – -teste

Utilizando o grep

GREP - **G**lobal **R**egular **E**xpression **P**rint, consiste de uma família de comandos (*grep*, *egrep*, *fgrep*), utilizados para buscar padrões em arquivos, que ecebem como entrada arquivos (ou dados provindos de pipes, entrada padrão) e retornam as linhas que "casam" com o padrão de busca informado pelo usuário.

Utilizando o grep

- Sintaxe: *grep [opções] <padrão> <arquivos/dados>*
- <padrão>: Padrão de busca.
- Comandos:
  - grep: Aceita como padrão de busca expressões regulares ou não.
  - egrep: Extended grep, deve ser utilizado somente quando for necessário a utilização de expressões regulares mais complexas (é mais lento).
  - fgrep: Fast grep, ideal para buscas simples que não envolvam expressões regulares, é o mais rápido da família.

Utilizando o grep

Exemplos:

- grep “/dev/*” /etc/fstab
- grep “swap” /etc/fstab
- grep -v “swap” /etc/fstab
- grep -H “^♯” /etc/fstab
- grep “^/\ |^♯” /etc/fstab
- grep –color “/dev/[a-zA-Z]\{3,\}[0-9]” /etc/fstab

- **cat** - Concatena arquivos (exibe conteúdo)
- **wc** - Conta caracteres, linhas, palavras
- **head** - Exibe início do arquivo
- **tail** - Exibe final do arquivo
- **cut** - Remove seções de cada linha de um arquivo
- **sort** - Ordenação
- **paste** - Junta arquivos

Exercícios

1. Exiba o conteúdo do arquivo /etc/fstab
2. Conte o número de linhas do arquivo /etc/fstab
3. Exiba somente as duas primeiras linhas do arquivos /etc/fstab (Dica: man head)
4. Exiba somente as duas ultimas linhas do arquivos /etc/fstab (Dica: man tail)
5. Execute os seguintes comandos no diretório shell:
   seq 1 10 > f1
   seq 10 -1 1 > f2
   Os arquivos f1 e f2 serão criados. Ordene o arquivo f1.
   Ordene novamente agora utilizando a opção -g.
6. Exiba a junção dos arquivos f1 e f2

Respostas

1. cat /etc/fstab
2. wc -l /etc/fstab
3. head -n 2 /etc/fstab
4. tail -n 2 /etc/fstab
5. sort f1
   sort -g f1
6. paste f1 f2

Bibliografia


Neves, Julio Cezar. Programação SHELL LINUX - 6ª edição. Brasport, 2006.


Bash, manual pages.

Até Logo!

 [1]

[1] "Programação Shell Script: dominando seu terminal", por Renê de Souza Pinto, é licensiado sob a Creative Commons Atribuição-Uso Não-Comercial 2.5 Brasil License.