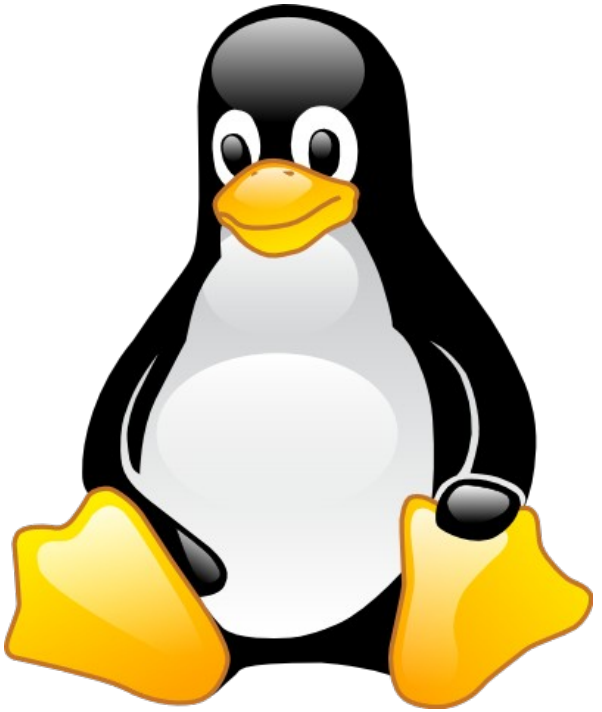
Linux System Administration 455

Aula 10 – 455



Aula 10 – 455

Funções do Shell

Analisar dados a partir do prompt (dados de entrada);
Interpretar comandos;
Controlar ambiente Unix-like (console);
Fazer redirecionamento de entrada e saída;
Execução de programas;
Linguagem de programação interpretada.

Aula 10 – 455

O uso da tralha (#)

A tralha ou jogo da velha (#) representa, em várias linguagens de programação, um comentário, o mesmo acontece com o Shell Script.

Um script em Shell é iniciado com a seguinte linha:
#!/bin/SHELL_EM_USO

Para o GNU/Linux:
#!/bin/bash

Esta linha acima indica o caminho (path) para o interpretador que será usado no script.

Aula 10 – 455

As crases são usadas para dar prioridade a um comando, veja um exemplo:

$ echo “A versão do kernel do `uname -o` é `uname -r`”

Saída:
A versão do kernel deste GNU/Linux é 2.6.15

Se você tirar as crases, veja a saída:
A versão do kernel deste uname -o é uname -r

Aula 10 – 455

Uma variável é representada por $ (cifrão).

Exemplo de variável:
guarda_roupa=camiseta
echo $guarda_roupa

Saída:
camiseta

Aula 10 – 455

Depois de editar um novo script, é necessário que modifiquemos a permissão deste arquivo, senão este não poderá ser executado, veja o porquê:

```
$ ls -l
-rw-r--r-- 1 leo users 0 2006-05-20 13:20 backup.sh

$ chmod +x backup.sh
$ ls -l
-rwxr-xr-x 1 leo users 0 2006-05-20 13:20 backup.sh
```

Para executar:

```
$ ./backup.sh
```

Aula 10 – 455

Operadores aritméticos:

+ Soma
- Subtração
* Produto
/ Divisão
% Resto da divisão

Aula 10 – 455

O comando expr

Para fazer um cálculo é necessário usar o comando expr.

Exemplos:

$ expr 20 + 05
$ expr 20 - 05
$ expr 20 * 05
$ expr 20 / 05
$ expr 20 % 05

Aula 10 – 455

Um parâmetro é representado por $n, onde n é a posição do carácter ou conjunto de caracteres.

$1 seria o primeiro caracter ou o primeiro conjunto de caracteres;
$2 seria o segundo caracter ou o segundo conjunto de caracteres.

Aula 10 – 455

Vamos supor que eu tenho um programa chamado "monte_nome" e eu quero exibir o nome montado após receber letra por letra. Veja o script:

```
#
# Script para montar nomes
# Este script recebe nove parâmetros
#

#!/bin/bash

echo $1 $2 $3 $4 $5 $6 $7 $8 $9

#
# Fim do script
#
```

$./monte_nome M A R I A (Note que entre cada parâmetro há um espaço)
Saída: MARIA

Aula 10 – 455

E o parâmetro $0? Alguém imagina qual seja o seu conteúdo?

O parâmetro $0 representa o nome do próprio programa.

Aula 10 – 455

Mas pense neste programa que fiz, é péssimo pois está limitado a nove caracteres apenas. É simples arrumar isso, veja:

```
#
# Script para montar nomes
# Este script recebe "n" parâmetros
#

#!/bin/bash

echo $*

echo Foram passados $# parametros

#
# Fim do script
#
```

```
$ ./monta_nome M A R I A D A S I L V A
```

Saída: MARIADASILVA

Aula 10 – 455

O $* recebe todos os parâmetros passados.

Aula 10 – 455

vi usuarios.sh

```
#!/bin/bash
echo “Lista completa dos usuários do sistema”
cut -f1 -d : /etc/passwd | sort | more
echo “O sistema possui `cat /etc/passwd | wc -l` usuários."
```

Aula 10 – 455

chmod +x usuarios.sh

Então vamos executar:

./usuarios.sh