INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Introdução a Sistemas Abertos

Shell Script

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Introdução

- O que é shell?
  - É o programa que permite ao usuário interagir com o sistema operacional através da linha de comando.
  - O shell possui estruturas de linguagem de programação como variáveis, estruturas de decisão e funções. Isso possibilita a automação de tarefas por meio da utilização de scripts.
- O que Script?
  - É um arquivo que guarda vários comandos e pode ser executado sempre que preciso.
  - Os comandos de um script são exatamente os mesmos que se digita no prompt da linha de comando.

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Criando Shell Script

- Utilize algum editor de texto para criar o arquivo.

```
aluno@debian:~$ vi shell.sh
```

- A primeira linha do script informa que o shell interpretará os comandos:

```
#!/bin/bash
```

- Após digitar a sequência de comandos, salve o arquivo é atribua permissão de execução.

```
aluno@debian:~$ chmod +x shell.sh
```

Caso esteja no mesmo diretório do script, execute utilizando um ./ antes do nome: **./script.sh**
Caso contrário digite o caminho completo: **/home/aluno/script.sh**

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Exemplo

```
#!/bin/bash

#eu sou um comentário
echo "hello world"
date
```

Para inserir um comentário em códigos de shell script basta inserir o # na frente da linha

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Variáveis

- São nas variáveis que os dados obtidos durante a execução do script serão armazenados. Para definir uma variável, basta usar o sinal de igual "=" e para ver seu valor, usa-se o “echo”.

- O bash reconhece uma variável quando ela começa com $, ou seja, a diferença entre ‘palavra’ e ‘$palavra’ é que a primeira é uma palavra qualquer, e a outra uma variável.

```
aluno@debian:~$ VARIAVEL=“Hello World"
aluno@debian:~$ echo $VARIAVEL
Hello World
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Variáveis

- Para direcionar a saída de um determinado comando para uma variável, basta escolher uma das seguintes sintaxes: **nome_variavel=$(comando)** ou **nome_variavel=`comando`**

```
#!/bin/bash
HOJE=$(date)
#eu sou um comentário
echo "hello world"
echo $HOJE
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Exemplo

- Crie um script que exiba a quantidade de arquivos em um diretório.

```
#!/bin/bash

total_arquivos=$(ls | wc -l)

echo “Existe $total_arquivos arquivos neste diretório.”
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Capturar Valores do Teclado

- É possível que você precise que o usuário defina uma variável durante a execução do comando.

- O comando **read**, dá uma pausa no script até que o usuário digite um valor e teclar enter.

```
#!/bin/bash

echo "Digite o valor da variavel: "
read VARIAVEL
echo “A variavel e $VARIAVEL"
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Parâmetros

- Os scripts podem receber dados diretamente via linha de comando.
- As variáveis passadas na linha do comando são definidas automaticamente, como $1 para o primeiro parâmetro, $2 para o segundo e assim po diante.
  - $@ ou $* - Todos os parâmetros a partir de $1.
  - $# - O número de parâmetros passados.
  - $$ - Pid do processo atual.
  - $0 - O nome do script.

Parâmetros

```
#!/bin/sh
# argumentos - mostra o valor das variáveis especiais

echo "O nome deste script é: $0"
echo "Recebidos $# argumentos: $*"
echo "O primeiro argumento recebido foi: $1"
echo "O segundo argumento recebido foi: $2
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Expressões Aritméticas

- O shell pode executar operações matemáticas. Para isso basta utilizar o bloco $((…)).

```
aluno@debian:~$ echo $((2*3))
6
aluno@debian:~$ echo $((2*3-2/2+3))
8
aluno@debian:~$ NUM=44
aluno@debian:~$ echo $((NUM*2))
88
aluno@debian:~$ NUM=$((NUM+1))
aluno@debian:~$ echo $NUM
45
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Execução Condicional

- O shel possui estruturas para tomada de decisão como o if e o case.
- O if realiza uma operação quando uma condição é atendida.
  - Sintaxe if:

```
if [ "$VARIAVEL" -gt 10 ]
then
   echo "é maior que 10"
else
   echo "é menor que 10"
fi
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Tabela de Operadores test

-eq	Igual
-ne	Diferente
-gt	Maior
-lt	Menor
-o	Ou
-d	Se for um diretório
-e	Se existir
-z	Se estiver vazio
-f	Se conter texto
-o	Se o usuário for o dono
-r	Se o arquivo pode ser lido
-w	Se o arquivo pode ser alterado
-x	Se o arquivo pode ser executado

Exemplo:

```
if [ -e $linux ]
then
  echo 'A variável $linux existe.'
else
  echo 'A variável $linux não existe.'
fi
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Execução Condicional

- O case utiliza múltiplas opções.
  - Sintaxe case:

```
#!/bin/bash

echo "Digite 1 ou 2"
read opcao;

case "$opcao" in
"1")
      echo "Você digitou 1"
;;
"2")
      echo "Você digitou 2"
;;
esac
exit
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Repetição Condicional

- O for executa uma ação repetidamente até que a condição seja atendida.
  - Sintaxe for:

```
for variavel in lista;
do
comandos
done
```

  - **variavel**: iterador para cada elemento da lista
  - **lista**: saída de comando ou variável com lista de elementos
  - **comando**: comandos que serão executados.

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Repetição Condicional

- O for executa uma ação repetidamente até que a condição seja atendida.
  - Sintaxe for:

```
#!/bin/bash

for contador in {1..4};
 do

   echo "Messagem exibida $contador vez".

   sleep 1;

 done
```

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Repetição Condicional

- O while executa uma ação repetidamente até que a condição seja falsa.
  - Sintaxe while:

```
#!/bin/bash

contador=1

while [ $contador -ne 4 ];

do

    echo "Messagem exibida $contador vez".
    sleep 1;
    ((contador=$contador+1))

done
```

Referências

- JARGAS, A. M. Introdução ao Shell Script. Disponível em: http://aurelio.net/shell/apostila-introducao-shell.pdf. Acesso em 05 de março de 2015.

- Programando em Shell Script. Disponível em: http://www.devin.com.br/shell_script/. Acesso em 05 de março de 2015.