INSTITUTO POLITECNICO
DE VIANA DO CASTELO
Escola Superior
de Tecnologia e Gestão

A Variável "Exit Status"

Departamento das Ciências Básicas e da Computação
Docente: Vítor Manuel Ferreira
ferreira@estg.ipvc.pt

Sumário

- Noção de “True” e “False”
- Executar um comando condicionalmente
- A solução com “if”
- Utilizando o comando “test”

Sumário

- **Noção de “True” e “False”**
- Executar um comando condicionalmente
- A solução com “if”
- Utilizando o comando “test”

A Variável "Exit Status"

- Quando um programa/comando UNIX termina, implicitamente (e não explicitamente como iremos ver) retorna um valor ao programa que o lançou (normalmente, a shell) informando-o se foi ou não executado com sucesso.
- Esse valor é um número e é chamado o "**exit status**" do programa/comando.

A Variável "Exit Status"

- Esse valor, o "exit status", é normalmente ignorado, quer pela shell quer pelo utilizador.
- No entanto, na construção de "shell scripts" este valor é muito importante.
- Normalmente, se o valor do "exit status" for **igual a 0** significa que o programa foi **executado com sucesso**, enquanto que se for **diferente de 0** significa que **ocorreu um erro**.

A Variável "Exit Status"

- Então coloca-se a seguinte questão: como examinar o valor do "**exit status**"?
- O valor do "exit status" do último programa/ comando executado é gravado na variável "?" e pode ser consultada a qualquer momento através do comando:
  - echo $?

A Variável "Exit Status"

- É de notar que, o valor desta variável é sistematicamente actualizada cada vez que um comando é executado (incluindo o comando *echo*)
- Vejamos:
  - ls; echo $?
  - ls dddd; echo $?; echo $?

Noção de "True" e "False"

- Deste modo, torna-se útil na construção de "shell scripts" pensar que:

* se o valor do "exit status" for 0, temos o equivalente ao termo lógico "true"

* e se for diferente de 0, temos o equivalente ao termo lógico "false"

Noção de “True” e “False”

- É de notar que, esta convenção aqui usada é exactamente o oposto ao que estamos habituados com outras linguagens de programação, como por exemplo em “C”
- Mais ainda, até existem comandos em UNIX chamados “true” e “false” que nos indicam exactamente esta nova convenção usada com o “exit status”.

Noção de “True” e “False”

- Façamos:
  - true; echo $?
  - false; echo $?

Sumário

- Noção de “True” e “False”
- Executar um comando condicionalmente
- A solução com “if”
- Utilizando o comando “test”

Sumário

- Noção de "True" e "False"
- **Executar um comando condicionalmente**
- A solução com "if"
- Utilizando o comando "test"

Executar um comando condicionalmente

- É sempre possível especificar em que condições um determinado comando numa "scritp" deve ser exucutado

- Tais condições são sempre explicitamente expressas em termos do "exit status" de outro comando. Isto é, considere a seguinte linha de comandos:

  - comando1 && comando2

Executar um comando condicionalmente (cont.)

- Significa que, o comando2 só será executado se o comando1 for executado com sucesso, i.e., com o valor do "exit status" igual a 0
- Outro exemplo: comando3 || comando4;
- significa que, o comando4 só será executado se o comando3 não executado com sucesso, sendo valor do "exit status" diferente de 0

Executar um comando condicionalmente (cont.)

- Por exemplo:
  - ls file1 && cp file1 /tmp
  - cp abc xyz && echo “O ficheiro foi copiado com sucesso”
  - diff fileA fileB || echo “Os ficheiros são diferentes”

Executar um comando condicionalmente (termo)

- É de notar que este tipo de condições são muito limitadas:
  - ✓ Só pode executar um comando se a condição se verificar (apesar de ser possível agrupar comandos)
  - ✓ Não se consegue especificar um segundo comando alternativo caso a condição não se verificar

Sumário

- Noção de "True" e "False"
- Executar um comando condicionalmente
- A solução com "if"
- Utilizando o comando "test"

Sumário

- Noção de “True” e “False”
- Executar um comando condicionalmente
- **A solução com “if”**
- Utilizando o comando “test”

A solução com “if”

- Muito mais poderoso e de mais fácil leitura:
  - if comando1

    then

        comando2

        comando3

    fi

A solução com "if" (cont.)

- Por exemplo:
  - if diff file1 file2 2> /dev/null

then

    echo "Os ficheiros são iguais"

    rm file2

fi

Sumário

- Noção de "True" e "False"
- Executar um comando condicionalmente
- A solução com "if"
- Utilizando o comando "test"

Sumário

- Noção de "True" e "False"
- Executar um comando condicionalmente
- A solução com "if"
- **Utilizando o comando "test"**

Utilizando o comando “test”

- A maior parte das linguagens de programação suportam a noção de “comparar” dois valores, duas variáveis ou uma variável e um valor

- Os valores podem ser comparados por forma a verificarmos se são iguais, diferentes, qual o maior ou o menor, etc

- A bash, nativamente, não suporta tais comparações, mas existe um comando programa da shell que o faz: “test”

Utilizando o comando “test”

- O comando “test” é usado da seguinte forma:

```
Ferreiras-Macbook:~ ferreira$ VAR_1=10
Ferreiras-Macbook:~ ferreira$ test $VAR_1 = 20
Ferreiras-Macbook:~ ferreira$ echo $?
1
Ferreiras-Macbook:~ ferreira$
```

- A única finalidade do comando “test” é devolver o valor do “exit status” de acordo com a condição testada
- Esse valor do “exit status” devolvido é consistente com a noção de “true” e “false”

Utilizando o comando “test”

- Por outras palavras, no exemplo dado, temos uma condição falsa
- Podemos assim usar o comando “test” com a expressão condicional “if” da seguinte forma:

```
Ferreiras-Macbook:~ ferreira$ if test $VAR_1 -gt $max
> then
> echo "Este valor é demasiado grande"
> fi
```

Sumário

- Noção de “True” e “False”
- Executar um comando condicionalmente
- A solução com “if”
- Utilizando o comando “test”

Bibliografia

- Ficha de trabalho no. 6