# UFCD 10806

Ficha 6

Fonte: webminal.org

Scripting introduction

Por Paul Cobbaut e este guia gratuito está disponível em http://linux-training.be

Shells como **bash** e **Korn** têm suporte para construções de programação que podem ser salvas como **scripts**. Esses **scripts**, por sua vez, tornam-se mais  **comandos shell**. Muitos comandos Linux são **scripts**. **Os scripts de perfil de usuário** são executados quando um usuário faz logon e **os scripts de inicialização** são executados quando um **daemon** é interrompido ou iniciado.

Isso significa que os administradores de sistema também precisam de conhecimentos básicos de **scripts** para entender como seus servidores e seus aplicativos são iniciados, atualizados, corrigidos, mantidos, configurados e removidos, e também para entender como um ambiente de usuário é construído.

O objetivo deste capítulo é dar-lhe informações suficientes para ser capaz de ler e entender scripts. Não para se tornar um escritor de scripts complexos.

Hello World

Assim como em todo curso de programação, começamos com um script hello\_world simples. O script a seguir produzirá **Hello World**.

echo Hello World

After creating this simple script in **vim** or with **echo**, you'll have to **chmod +x hello\_world** to make it executable. And unless you add the scripts directory to your path, you'll have to type the path to the script for the shell to be able to find it.

[paul@RHEL4a ~]$ echo echo Hello World > hello\_world.sh

Ou

[]$ nano hello\_world.sh

[paul@RHEL4a ~]$ chmod +x hello\_world

[paul@RHEL4a ~]$ ./hello\_world

Hello World

[paul@RHEL4a ~]$

she-bang

Let's expand our example a little further by putting **#!/bin/bash** on the first line of the script. The **#!** is called a **she-bang** (sometimes called **sha-bang**), where the **she-bang** is the first two characters of the script.

#!/bin/bash

echo Hello World

You can never be sure which shell a user is running. A script that works flawlessly in **bash** might not work in **ksh**, **csh**, or **dash**. To instruct a shell to run your script in a certain shell, you can start your script with a **she-bang** followed by the shell it is supposed to run in. This script will run in a bash shell.

#!/bin/bash

echo -n hello

echo A bash subshell `echo -n hello`

This script will run in a Korn shell (unless **/bin/ksh** is a hard link to **/bin/bash**). The **/etc/shells** file contains a list of shells on your system.

#!/bin/ksh

echo -n hello

echo a Korn subshell `echo -n hello`

comment

Let's expand our example a little further by adding comment lines.

#!/bin/bash

#

# Hello World Script

#

echo Hello World

variables

Here is a simple example of a variable inside a script.

#!/bin/bash

#

# simple variable in script

#

var1=4

echo var1 = $var1

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer screen

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.



A screen shot of a computer

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.