Práctica 2: El shell como lenguaje de programación

Moisés García Villaneuva

23 de Agosto de 2012

1. Variables

Al igual que otros lenguajes de programación, en la programación en shell existen las variables, que son nombres simbólicos para zonas de memoria que almacenan datos que nos interesan. Pero al contrario que los lenguajes de alto nivel, las variables de los scripts del shell no tienen tipo, o quizás sería más apropiado decir que tienen un tipo único y permanente: a todos las variables se les trata como ristras de caracteres.

1.1. Referenciar variables

Referenciar variables

Las variables del shell no se declaran, y siempre están inicializadas con una ristra vacía o nula. Nótese que esto no es lo mismo que contener un espacio. Una ristra vacía o nula es una ristra que no contiene ningún caracter.

Hay que tener cuidado al asignar valores a las variables, ya que no se debe dejar ningún espacio entre el signo de asignación (el '=') y la variable o el valor asignado (algo que, por otra parte, es una muy buena costumbre cuando da igual).

Para referirse a las variables, hay que utilizar el signo dólar ($) antes del nombre de ésta, siempre que nos refiramos a ellas para consultar su valor (si asignamos un valor a la variable, o utilizamos la orden read, que escribe en ella, NO hay que poner el signo de dólar). Si nos olvidamos del signo dólar, y hacemos algo parecido a:

```
y=hola
x=y
```

Nos encontraremos con la desagradable sorpresa de que el valor de `x` es el carácter `y`, y no los caracteres hola, como quizás pretendiéramos. Para hacer la asignación correctamente, tendríamos que haber escrito:

```
y=hola
x=$y
```

Como en todas o al menos la mayoría de las cosas en UNIX, los nombres son sensibles a mayúsulas y minúsculas, es decir, que no es lo mismo `y` que `Y`.

1.2. Comillas

En las operaciones con la shell distinguimos tres tipos de comillas con distintas funcionalidades: las comillas dobles, las comillas simples y las comillas invertidas, el acento grave francés. A continuación describimos las funcdiones:

- ' Engloban un literal. La shell no trata de interpretar nada de lo que haya comprendido entre estas comillas.
- " La shell expande las variables que haya especificadas sustituyendo su nombre por el contenido. Para imprimir un $ es necesario protegerlo con una en otras palabras escaparlo.
- ` La shell intenta ejecutar lo que haya comprendido entre estas comillas.

Ejemplo: Asignamos una valor a una variable:

```
$  AA="ls -la"
```

y observamos la diferencia entre:

```
echo '$AA'
echo "$AA"
echo `$AA`
```

En bash podemos sustituir las comillas invertidas por el operador $().

http://dns.bdat.net/shell/x4657.html