

# Sesión 13: Mandatos, argumentos y variables

Programación para Sistemas

---

Ángel Herranz

Otoño 2018

Universidad Politécnica de Madrid

# En capítulos anteriores. . .

**Sesión 11:** Contacto Bash

**Sesión 12:** Practicando Bash

# En el capítulo de hoy...

- Mandatos sencillos
- Un poquito de **piping**
- Variables de entorno
- Comandos útiles

# Mandatos, argumentos y variables de entorno

```
$ mandato [arg1 [arg2 [...]]]
```

# Mandatos, argumentos y variables de entorno

```
$ mandato [arg1 [arg2 [...]]]
```

```
$ ls -al / ~
```

# Mandatos, argumentos y variables de entorno

```
$ mandato [arg1 [arg2 [...]]]
```

```
$ ls -al / ~
```

```
$ variable=valor mandato [arg1 [arg2 [...]]]
```

# Mandatos, argumentos y variables de entorno

```
$ mandato [arg1 [arg2 [...]]]
```

```
$ ls -al / ~
```

```
$ variable=valor mandato [arg1 [arg2 [...]]]
```

```
$ MAX_OUTPUT=100 ./secuencia -30 0 -3
```

# Mandatos, argumentos y variables de entorno

```
$ mandato [arg1 [arg2 [...]]]
```

```
$ ls -al / ~
```

```
$ variable=valor mandato [arg1 [arg2 [...]]]
```

```
$ MAX_OUTPUT=100 ./secuencia -30 0 -3
```

 Sin espacios alrededor de =



# Algunos mandatos conocidos: cd, ls, pwd

- ¿Qué hace ls?

⌚ 1'  man ls

# Algunos mandatos conocidos: cd, ls, pwd

- ¿Qué hace ls?

⌚ 1'  man ls

- ¿Qué hace cd?

# Algunos mandatos conocidos: cd, ls, pwd

- ¿Qué hace ls?

⌚ 1'  man ls

- ¿Qué hace cd?

 man **cd**

# Algunos mandatos conocidos: cd, ls, pwd

- ¿Qué hace ls?

🕒 1'  man ls

- ¿Qué hace cd?

 man **cd**  ¿Qué ocurre?

# Algunos mandatos conocidos: cd, ls, pwd

- ¿Qué hace ls?

🕒 1'  man ls

- ¿Qué hace cd?

 man **cd**  ¿Qué ocurre?

 Nombrado de ficheros y directorios

- **Absoluto**: el nombre empieza por /
- **Relativo**: el nombre **no** empieza por / y se convierte en absoluto concatenándolo al *working directory*.
- Elementos especiales: ~, ., ...



## Explorar el sistema de ficheros

- Empezar en / (**cd /**)
- Ejecutar `ls -al`
- Entrar en cada directorio y ejecutar `ls -al`
- Moverse a ~ (**cd ~**)
- Ejecutar `ls -al`
- Entrar en cada directorio y ejecutar `ls -al`

## Explorar el sistema de ficheros

- Empezar en / (**cd /**)
- Ejecutar `ls -al`
- Entrar en cada directorio y ejecutar `ls -al`
- Moverse a ~ (**cd ~**)
- Ejecutar `ls -al`
- Entrar en cada directorio y ejecutar `ls -al`

🕒 1'  <http://refspecs.linuxfoundation.org/fhs.shtml>

- Siempre a mano: `man bash`
- `cd` es un *built in command* de Bash
- Los mandatos de Bash pueden ser  
programas  
o  
built in commands
- Explorar el manual: `man bash` y `man PROGRAMA`



# ¿Dónde están los programas?


- Los programas están en el sistema de ficheros

 `which ls`

 ¿Y si hay dos programas con el mismo nombre?

## Variable de entorno **PATH**

- Un **path** es una lista de directorios<sup>1</sup>

 Mira y cambia el PATH

```
$ echo $PATH
```

```
$ ls
```

```
$ which ls
```

```
$ PATH=
```

```
$ echo $PATH
```

```
$ ls
```

```
$ which ls
```

---

<sup>1</sup>Separadas por el caracter ":" en Unix

# echo

🕒 1' 🔍 man **echo**

# echo

🕒 1' 🔍 man **echo**

💻 Ejecutar y *diseccionar*

```
$ echo Fíjate en los    espacios
```

```
$ echo "En un lugar de la mancha..." > quijote.txt
```

# echo

🕒 1' 🔍 man **echo**

💻 Ejecutar y *diseccionar*

\$ **echo** Fíjate en los      espacios

\$ **echo** "En un lugar de la mancha..." > quijote.txt

🏠 ¿Te atreves a programar echo en C?

🕒 1' 🔍 man **echo**

💻 Ejecutar y *diseccionar*

\$ **echo** Fíjate en los espacios

\$ **echo** "En un lugar de la mancha..." > quijote.txt

🏠 ¿Te atreves a programar echo en C?

💬 ¿Qué es echo? ¿Dónde está? ¿Por qué aparece en negrita en estas transparencias? ¿Has probado which? ¿Has mirado en man bash?


- ¿Qué hace cat?

🕒 1' 🔍 man **cat**

# cat i

- ¿Qué hace cat?

🕒 1'  man **cat**

 Ejecutar y *diseccionar*

```
$ cat quijote.txt
```

```
$ cat
```

 ¿Qué ocurre?

# cat i

- ¿Qué hace cat?

🕒 1' 🔍 man **cat**

💻 Ejecutar y *diseccionar*

```
$ cat quijote.txt
```

```
$ cat
```

💬 ¿Qué ocurre? Prueba a escribir

Los animales son felices mientras  
tengan salud y suficiente comida.

Ctrl-d



## Ejecutar y *disecccionar*

```
$ cat < quijote.txt
```

```
$ cat > felicidad.txt
```

Los animales son felices mientras  
tengan salud y suficiente comida.

Ctrl-d

```
$ cat quijote.txt felicidad.txt
```

```
$ cat quijote.txt felicidad.txt > dos_libros.txt
```

⌚ 2' 🔍 ¿Qué es read?

⌚ 2' 🔍 ¿Qué es read?

💻 Ejecutar y explorar:

```
$ read -p "¿Cuántos anos tienes?" EDAD
```

- *¿Crawling?*

---

<sup>2</sup>Instalado en triqui, instalable en Ubuntu con `apt-get install curl`

# Crawling i

- ¿Crawling?
- Usaremos el programar **cURL**<sup>2</sup>: `man curl`

```
$ curl http://www.fi.upm.es
```

```
$ curl http://www.fi.upm.es | grep href
```

---

<sup>2</sup>Instalado en triqui, instalable en Ubuntu con `apt-get install curl`

# Crawling i

- ¿Crawling?
- Usaremos el programar **cURL**<sup>2</sup>: `man curl`

```
$ curl http://www.fi.upm.es
```

```
$ curl http://www.fi.upm.es | grep href
```

**Q** ¿Qué es grep?

---

<sup>2</sup>Instalado en triqui, instalable en Ubuntu con `apt-get install curl`

# Crawling i

- ¿Crawling?
- Usaremos el programar **cURL**<sup>2</sup>: `man curl`

```
$ curl http://www.fi.upm.es
```

```
$ curl http://www.fi.upm.es | grep href
```

## Q ¿Qué es grep?

- *Crawling*:

```
$ curl -s http://www.fi.upm.es | grep -Po '(?<=href=")[^"]*(?=")'
```

---

<sup>2</sup>Instalado en triqui, instalable en Ubuntu con `apt-get install curl`

- *Almost magic:*

```
$ curl -s http://www.fi.upm.es |  
> grep -Po '(?<=href=")[^"]*(?=")' |  
> sort |  
> uniq
```

 man sort y man uniq



# Algo más útil: MP3 de youtube i

- <https://www.youtube.com/watch?v=ukKQw578Lm8>
- Necesitaremos dos programas
- *youtube-dl* - *download videos from youtube.com or other video platforms*
- *ffmpeg* - *ffmpeg video converter*
- Necesitaremos la versión más actual de *youtube-dl*:

```
$ cd tmp
$ curl -L https://yt-dl.org/downloads/latest/youtube-dl -o youtube-dl
$ chmod +x youtube-dl
$ ls -l youtube-dl
```

# Algo más útil: MP3 de youtube ii

- Descargamos el video

```
$ ./youtube-dl https://www.youtube.com/watch?v=ukKQw578Lm8
```

- Eso ha generado el fichero

```
TheLogicalSongporRogerHodgson-Letra.-ukKQw578Lm8.webm
```

- Guardamos el nombre sin extensión en una variable:

```
$ VIDEO="The Logical Song por Roger Hodgson -Letra.-ukKQw578Lm8"
```

- Lo pasamos a MP3

```
$ ffmpeg -i "$VIDEO.webm" -vn -ab 128k -ar 44100 -y "$VIDEO.mp3"
```

# ¿Por qué tanto entusiasmo?

Se puede  
automatizar

# ¿Por qué tanto entusiasmo?

Se puede  
automatizar en  
Scripts