

Sesión 10: Contacto Bash

Programación para Sistemas

Ángel Herranz

2020-2021

Universidad Politécnica de Madrid

Sesión 0: *Bash Crash Course*

Sesión 0: *Bash Crash Course*

★ <https://guide.bash.academy/>
¿Quién hizo esta guía?

En el capítulo de hoy...




- Sesión de programación en C para empezar a tocar...
 - Conceptos de sistemas operativos:
Ficheros, programas, procesos, variables de entorno, entrada/salida, invocación, estado de terminación, etc.
 - El juego de la expansión

Durante las siguientes semanas:

En el capítulo de hoy...

- Sesión de programación en C para empezar a tocar...
 - Conceptos de sistemas operativos:
Ficheros, programas, procesos, variables de entorno, entrada/salida, invocación, estado de terminación, etc.
 - El juego de la expansión

Durante las siguientes semanas:

-  Repasar transparencias sesión 0
-  Cuaderno de UNIX en github
-  <https://devhints.io/bash>

- Antes de *comenzar* con Bash necesitamos tener claros algunos **conceptos relacionados con el sistema operativo**
- **Ficheros, programas, procesos, variables de entorno, entrada/salida, invocación, estado de terminación, etc.**
- Lo vemos una tarea de C de **cursos pasados**: delreves¹

 **No** es una práctica de este año

- delreves invierte cada línea
- ¡*Nos vamos a triqui3.fi.upm.es!*

¹Sólo hacemos el esqueleto ;).

delreves (a triqui3.fi.upm.es)

```
$ whoami
```

```
angel
```

```
$ hostname
```

```
T440
```

```
$ ssh aherranz@triqui3.fi.upm.es
```

```
Last login: Tue Nov 20 09:31:28 2018 from sri33.sri.fi.upm.es
```

```
$ whoami
```

```
aherranz
```

```
$ hostname
```

```
triqui3.fi.upm.es
```

Línea de comandos (*command line*), invocación, programa, proceso, salida estándar

delreves (~pps/pub/tarea-2.1.2018-2019.tar.gz)

❓ ¿Qué es ~pps/pub/tarea-2.1.2018-2019.tar.gz?

delreves (~pps/pub/tarea-2.1.2018-2019.tar.gz)

❓ ¿Qué es ~pps/pub/tarea-2.1.2018-2019.tar.gz?

Un nombre de un fichero

Vamos a diseccionarlo

delreves (~pps/pub/tarea-2.1.2018-2019.tar.gz)

❓ ¿Qué es ~pps/pub/tarea-2.1.2018-2019.tar.gz?

Un nombre de un fichero

Vamos a diseccionarlo

❓ ¿Qué es ~pps?

delreves (~pps/pub/tarea-2.1.2018-2019.tar.gz)

❓ ¿Qué es ~pps/pub/tarea-2.1.2018-2019.tar.gz?

Un nombre de un fichero

Vamos a diseccionarlo

❓ ¿Qué es ~pps?

*Un nombre de un directorio, en concreto el directorio
home del usuario pps*

delreves (~pps/pub/tarea-2.1.2018-2019.tar.gz)

❓ ¿Qué es `~pps/pub/tarea-2.1.2018-2019.tar.gz`?

Un nombre de un fichero

Vamos a diseccionarlo

❓ ¿Qué es `~pps`?

Un nombre de un directorio, en concreto el directorio

home del usuario pps

 **echo** `~pps`

delreves (~pps/pub/tarea-2.1.2018-2019.tar.gz)

❓ ¿Qué es `~pps/pub/tarea-2.1.2018-2019.tar.gz`?

Un nombre de un fichero

Vamos a diseccionarlo

❓ ¿Qué es `~pps`?

*Un nombre de un directorio, en concreto el directorio
home del usuario pps*

 **echo** `~pps`

💬 ¿Por qué `/home/fi/dep/pps`? ¿Qué es?

*El nombre directorio home del usuario pps (nombre
de fichero absoluto)*

💬 ¿Qué es `~aherranz`? ¿Y `~y160215`? ¿Y `~`?

delreves (copiar a ~)

- Vamos a copiarnos ese fichero a *nuestra home*.

 A *home*: ¿cómo?

²También se les llama *tokens*.

delreves (copiar a ~)

- Vamos a copiarnos ese fichero a *nuestra home*.

 A *home*: ¿cómo?

```
$ cd
```

```
$ pwd
```

```
/home/fi/personal/dlsiis/aherranz
```

²También se les llama *tokens*.

delreves (copiar a ~)


- Vamos a copiarnos ese fichero a *nuestra home*.

 A *home*: ¿cómo?

```
$ cd
```

```
$ pwd
```

```
/home/fi/personal/dlsiis/aherranz
```

 Haz un `ls -al` y entiende lo que dice, línea por línea

²También se les llama *tokens*.

delreves (copiar a ~)

- Vamos a copiarnos ese fichero a *nuestra home*.

 A *home*: ¿cómo?

```
$ cd  
$ pwd  
/home/fi/personal/dlsiis/aherranz
```

 Haz un `ls -al` y entiende lo que dice, línea por línea

 `cp_~pps/pub/tarea-2.1.2018-2019.tar.gz_.`
Vamos a diseccionarlo (espacios resaltados con `..._...`)

²También se les llama *tokens*.

delreves (copiar a ~)

- Vamos a copiarnos ese fichero a *nuestra home*.

📁 A *home*: ¿cómo?

```
$ cd  
$ pwd  
/home/fi/personal/dlsiis/aherranz
```

📁 Haz un `ls -al` y entiende lo que dice, línea por línea

❓ `cp_~pps/pub/tarea-2.1.2018-2019.tar.gz_.`

Vamos a diseccionarlo (espacios resaltados con `..._.`)

- ¿Cuántas *palabras*² hay?

²También se les llama *tokens*.

delreves (copiar a ~)

- Vamos a copiarnos ese fichero a *nuestra home*.

📁 A *home*: ¿cómo?

```
$ cd
```

```
$ pwd
```

```
/home/fi/personal/dlsiis/aherranz
```

📁 Haz un `ls -al` y entiende lo que dice, línea por línea

❓ `cp_~pps/pub/tarea-2.1.2018-2019.tar.gz_.`

Vamos a diseccionarlo (espacios resaltados con `..._...`)

- ¿Cuántas *palabras*² hay? ¿Qué es `cp`? ¿Y `..`?

²También se les llama *tokens*.

delreves (copiar a ~)

- Vamos a copiarnos ese fichero a *nuestra home*.

 A *home*: ¿cómo?

```
$ cd
```

```
$ pwd
```

```
/home/fi/personal/dlsiis/aherranz
```

 Haz un `ls -al` y entiende lo que dice, línea por línea

 `cp_~pps/pub/tarea-2.1.2018-2019.tar.gz_.`

Vamos a diseccionarlo (espacios resaltados con `..._...`)

- ¿Cuántas *palabras*² hay? ¿Qué es `cp`? ¿Y `..`?

 Haz un `ls -al` ahora

²También se les llama *tokens*.

Explora y pregunta dudas

- Ejecuta **cd** .., luego **pwd** y luego **ls** -a! varias ocasiones

delreves (descomprimir en ~)

- Deberías **estar en tu home** (**cd**, **pwd** y **ls** si no estás seguro) y **no tener un directorio pps** (si ya existe puedes borrarlo con **rm** o renombrarlo con **mv**)

delreves (descomprimir en ~)

- Deberías **estar en tu home** (**cd**, **pwd** y **ls** si no estás seguro) y **no tener un directorio pps** (si ya existe puedes borrarlo con **rm** o renombrarlo con **mv**)
- ❓ **tar_xvfz_tarea-2.1.2018-2019.tar.gz**
- ¿Cuántas *palabras* hay?

delreves (descomprimir en ~)

- Deberías **estar en tu home** (**cd**, **pwd** y **ls** si no estás seguro) y **no tener un directorio pps** (si ya existe puedes borrarlo con **rm** o renombrarlo con **mv**)
- ❓ **tar_xvfz_tarea-2.1.2018-2019.tar.gz**
- ¿Cuántas *palabras* hay? ¿Qué es **tar**? ¿Y **xvfz** y **tarea-2.1.2018-2019.tar.gz**?

delreves (descomprimir en ~)

- Deberías **estar en tu home** (**cd**, **pwd** y **ls** si no estás seguro) y **no tener un directorio pps** (si ya existe puedes borrarlo con **rm** o renombrarlo con **mv**)

❓ **tar_xvzf_tarea-2.1.2018-2019.tar.gz**

- ¿Cuántas *palabras* hay? ¿Qué es **tar**? ¿Y **xvzf** y **tarea-2.1.2018-2019.tar.gz**?

```
$ tar xvzf tarea-2.1.2018-2019.tar.gz
pps/tarea-2.1.2018-2019/
pps/tarea-2.1.2018-2019/auxiliar.c
pps/tarea-2.1.2018-2019/autores.txt
pps/tarea-2.1.2018-2019/bitacora.txt
pps/tarea-2.1.2018-2019/auxiliar.h
pps/tarea-2.1.2018-2019/delreves.c
pps/tarea-2.1.2018-2019/ejemplo.c
pps/tarea-2.1.2018-2019/Makefile
pps/tarea-2.1.2018-2019/pruebas.sh
```

delreves (¡a programar!)

- Vamos hasta el directorio `~/pps/tarea-2.1.2018-2019`

 ¿Diferencias?

`~pps/pub/tarea-2.1.2018-2019.tar.gz`

vs.

`~/pps/tarea-2.1.2018-2019`

delreves (¡a programar!)

- Vamos hasta el directorio `~/pps/tarea-2.1.2018-2019`

💬 ¿Diferencias?

`~pps/pub/tarea-2.1.2018-2019.tar.gz`

vs.

`~/pps/tarea-2.1.2018-2019`

- Haz un `ls`
- ¡Tenemos un Makefile!

💻 **cat** Makefile

💻 **make**

- ¡No compila porque `delreves.c` está vacío!

- Un programa que no hace nada, salvo **terminar bien**:

```
int main() {  
    return 0;  
}
```

📄 make + ejecución:

```
$ make  
Autores:  
[...]  
gcc -Wall -g delreves.c -o delreves auxiliar.o -lm  
$ ./delreves  
$ |
```

📄 echo \$?

- Un programa que no hace nada, salvo **terminar bien**:

```
int main() {  
    return 0;  
}
```

📁 make + ejecución:

```
$ make  
Autores:  
[...]  
gcc -Wall -g delreves.c -o delreves auxiliar.o -lm  
$ ./delreves  
$ |
```

📁 **echo** \$? ← la **variable Bash** "\$?" contiene estado de terminación o **exit status** del último mandato ejecutado

Herranz, ¿cómo lo haces? i

Q ¿Puedes editar en tu máquina local y luego copiarlo a triqui?

A Sí, usando scp. scp acepta dos argumentos, el fichero local y el fichero remoto (o al revés cuando quieres traerte cosas del servidor)

usuario@máquina:nombre_fichero

(si el nombre es relativo lo es al respecto del home)

Herranz, ¿cómo lo haces? ii

Q ¿Puedes ejecutar desde tu máquina local un mandato en una máquina remota?

A Sí, con ssh (man ssh :))

```
ssh aherranz@triqui3 ls -al
```

Q ¿¡Puedes editar desde tu máquina local un fichero remoto!?

A Sí, algunos editores de texto entienden nombres de ficheros en máquinas remotas. Emacs, por ejemplo.

Herranz, ¿cómo lo haces? iv

Q ¿Cómo consigues que triqui no te pida password?

A Generas una pareja de claves con ssh-keygen y copias la clave pública a triqui

- Un programa que da **error** cuando el número de **parámetros no es el apropiado**:

```
#include <stdio.h>
int main(int argc, char *argv[]) {
    if (argc > 2) {
        fprintf(stderr, "ERROR\n");
        return -1;
    }
    return 0;
}
```



Prueba estas invocaciones (**echo \$?**):

```
$ ./delreves
$ ./delreves -h
$ ./delreves a b
```

- Un programa que da un **mensaje de ayuda** cuando se **invoca con -h** (y termina bien):

```
#include <string.h>
...
}
if (argc == 2 && strcmp(argv[1], "-h") == 0) {
    printf("Uso: %s [FICHERO]\n", argv[0]);
    return 0;
}
return 0;
}
```



Prueba las mismas invocaciones:

```
$ ./delreves
$ ./delreves -h
$ ./delreves a b
```

- Un programa que abre el fichero adecuado³ y da un error si no se pudo abrir:


```
...  
int main(int argc, char *argv[]) {  
    FILE *fichero;  
    ...  
    if (argc == 2)  
        fichero = fopen(argv[1], "r");  
    else  
        fichero = stdin;  
    if (fichero == NULL) {  
        fprintf(stderr, "El fichero no se pudo abrir\n");  
        return -1;  
    }  
    return 0;  
}
```

³man fopen

 Prueba las invocaciones:

```
$ ./delreves  
$ ./delreves -h  
$ ./delreves a b  
$ ./delreves supercalifrasgilisticoepialidoso  
$ ./delreves delreves.c
```

- Un programa que lee líneas y las escribe.

 `man stdin`, `man fopen` y `man fgets`

- Un programa que lee líneas y las escribe.

 man stdin, man fopen y man fgets

```
extern FILE *stdin;
```

```
FILE *fopen(const char *pathname, const char *mode);
```

```
char *fgets(char *s, int size, FILE *stream);
```

- Recordemos

```
FILE *fichero;
```

```
...
```

```
if (argc == 2)
```

```
    fichero = fopen(argv[1], "r");
```

```
else
```

```
    fichero = stdin;
```

- Un programa que lee líneas y las escribe.

- Un programa que lee líneas y las escribe.

```
char linea[2049];  
...  
while (fgets(linea, 2048, fichero) != NULL) {  
    printf("%s", linea);  
}
```

- 📄 Invertir la línea antes de imprimirla y probar el programa: entrada estándar (teclado), desde fichero, con redirección

```
$ ./delreves  
$ ./delreves mifichero.txt  
$ ./delreves < mifichero.txt
```

delreves.c vii

- Aprendemos a usar la biblioteca auxiliar.h:

```
...  
#include "auxiliar.h"  
int main(int argc, char *argv[]) {  
    FILE *fichero;  
    /* argv0 es una variable global de auxiliar */  
    argv0 = argv[0];  
    if (argc > 2) {  
        Error(EX_USAGE, "Error de uso");  
    }  
    ...  
    return 0;  
}
```





Prueba ahora:

```
$ ./delreves a b  
./delreves: Error(EX_USAGE), uso incorrecto del mandato. "Success"  
./delreves+ Error de uso  
$ |
```

 Explora la biblioteca *auxiliar*: *auxiliar.h* y *auxiliar.c*

```
/* Declaraciones en auxiliar.h */  
extern char * argv0;  
void Error(int exitval, char * fmt, ...);  
/* Definiciones y usos en auxiliar.c */  
char * argv0 = "<argv0>";  
fprintf(stderr, "%s: Error(%s), %s. \"%s\"\\n", argv0,...);
```

 Busca *sysexits.h* (*locate* *sysexits.h*)

 Modificar *delreves.c* para que el mensaje de error cuando el fichero no se puede abrir sea más adecuado (utilizar la biblioteca *auxiliar*)