Salida de un script

Para coger la salida de un script, se hace con el signo flecha (>) y doble flecha (»)

```
  - > Crea el fichero de nuevo. Si existe, lo borra
  - >> Añade al final del fichero la salida
```

Ejemplo:

```
# ls
fichero1.txt  fichero2.txt  fichero3.txt  fichero4.txt
```

Lo redirejimos:

```
# ls > ls.txt
#
```

```
# cat ls.txt
fichero1.txt
fichero2.txt
fichero3.txt
fichero4.txt
ls.txt
```

Hay otra salida que es el error. Es decir, si hacemos ls de un fichero que no existe nos dará error:

```
# ls jur.txt
ls: cannot access 'jur.txt': No such file or directory
```

Si hacemos la redirección, nos aparece en pantalla y no lo graba en el fichero. El fichero ls.log está vacio:

```
# ls jur.txt > ls.log
ls: cannot access 'jur.txt': No such file or directory

# cat ls.log
#
```

La **salida error** (stderr) es la 2 y la **salida estándar** (stdout) es la 1. Podemos redirigir la salida de error a otro fichero:

```
# ls jur.txt > ls.log 2>ls_error.log
#
```

```
# cat ls_error.log
ls: cannot access 'jur.txt': No such file or directory
```

Para juntar todo en el mismo log:

```
# ls jur.txt > ls.log 2>&1
#

# cat ls.log
ls: cannot access 'jur.txt': No such file or directory
```

Nombre de Ficheros Una buena idea es poner fecha en las líneas de código o en un los nombres. Con date:

```
date +%Y%m%d_%H%M%S
20210222_101819
```

Crontab

El crontab usa path relativos, variables de entornos propias y a veces no sabemos porque falla. Es interesante redirigir toda la salida a un fichero de log al principio para controlarlo:

```
crontab -l
```

```
*/1  *  *  *  *  script.sh >> /home/usuario/crontab.log 2>&1
```