

# **Redireccionamiento de entrada y salida**



# stdin, stdout, stderr

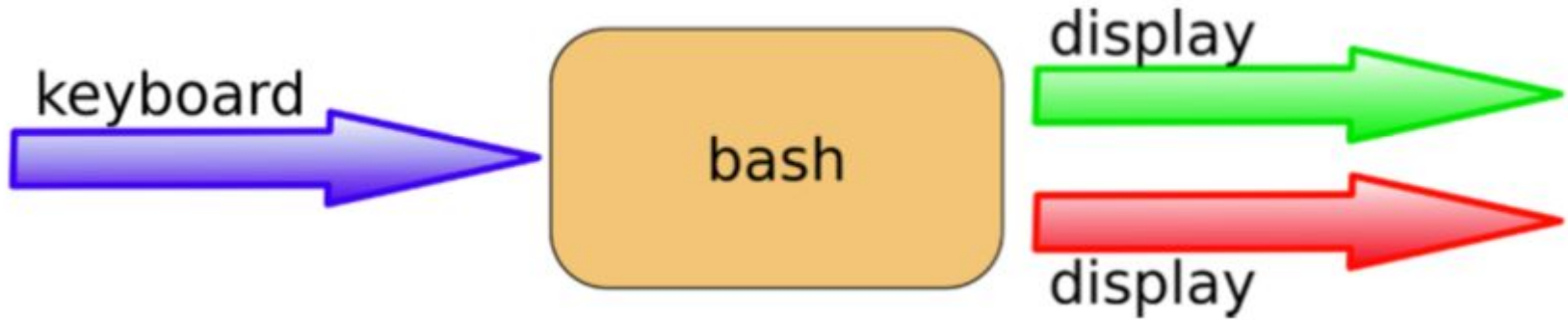
La shell de bash tiene tres flujos básicos. Toma la entrada de stdin (flujo 0), envía la salida a stdout (flujo 1) y dirige los errores a stderr (flujo 2)





# stdin, stdout, stderr

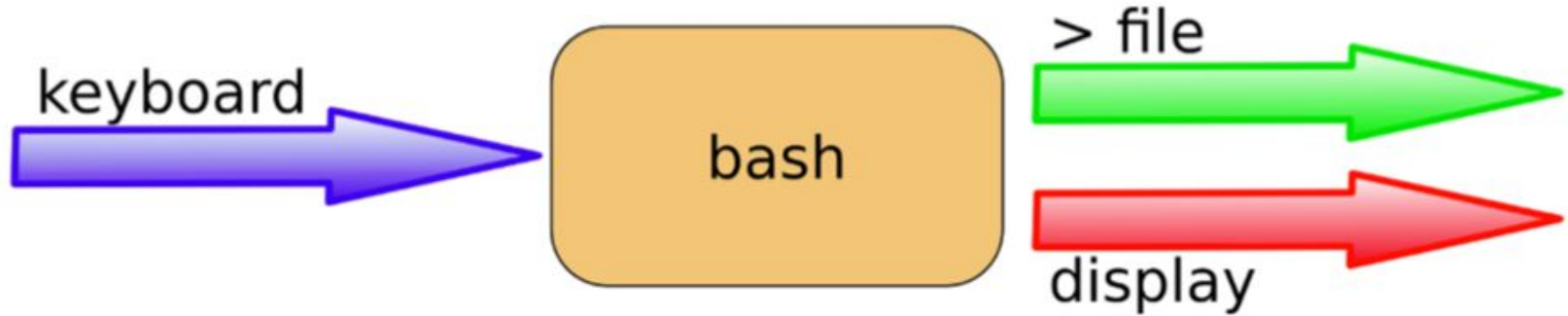
La mayoría de las veces el teclado sirve stdin, mientras que stdout y stderr van a pantalla





Los usuarios experimentados saben que puede ser bastante útil separar los errores de la salida.

# 1. Redirección de salida a archivos





**> stdout**

La salida puede ser redirigida con el símbolo de **mayor que**.

Ejemplo:

```
echo winter is coming > got8.txt
```

La notación **>** es una abreviación de **1>** (recordando que stdout es el flujo 1)

```
echo winter is coming 1> got8.txt
```



## ¿Cómo lo interpreta la shell?


La shell resuelve el redireccionamiento antes de que el argumento cero se ejecute

`echo hola > saludo.txt`

La shell solo toma dos argumentos

`Argumento_cero = echo`

`Argumento_uno = hola`



## **Los archivos de salida se borran**

Mientras se escanea una línea la shell detectará el símbolo > y limpiará el archivo de salida.

Dado que esto sucede antes de resolver el argumento cero, incluso si el comando falla, el archivo se borrará





# **noclobber**

El borrado de archivos se puede prevenir utilizando la opción `noclobber`

```
cat hola.txt
```

```
set -o noclobber
```

```
echo mundo > saludo.txt
```

```
set +o noclobber
```



## Anulando noclobber

Se puede anular la opción noclobber utilizando los símbolos >|

```
set -o noclobber
```

```
echo mundo > saludo.txt
```

```
echo mundo >| saludo.txt
```

```
set +o noclobber
```

```
cat saludo.txt
```



**>> file**

Utilizar el símbolo >> para concatenar a un archivo de salida.

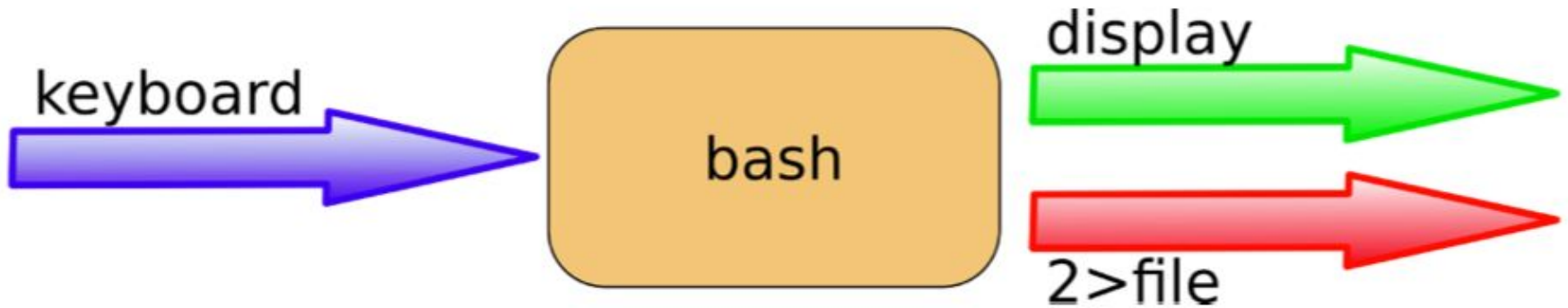
```
echo hola mundo > saludo.txt
```

```
cat saludo.txt
```

```
echo adios mundo > saludo.txt
```

```
cat saludo.txt
```

## 2. Redirección de error a archivos





## **2> archivo**

Se puede redireccionar el error utilizando los símbolos **2>**

Ejemplo:

```
find / > all_files.txt 2> /dev/null
```

Esto es útil para evitar saturar la pantalla con mensajes.



**2>&1**

Con esta instrucción se puede forzar que **stderr** vaya a **stdout**

Para redireccionar **stderr** y **stdout** al mismo archivo usando **2>&1**

```
find / > all_files_all_errors 2>&1
```





**1>&2**

Con esta instrucción se puede forzar que **stdout** vaya a **stderr**

Para redireccionar **stdout** y **stderr** al mismo archivo usando **1>&2**

```
find / 2> all_files_all_errors 1>&2
```



## El orden es importante

```
find / > all_files_all_errors 2>&1
```

Redirecciona stdout y stderr al archivo dirlist

```
find / 2>&1 > all_files_all_errors
```

Redirecciona solo salida estándar al archivo dirlist porque stderr se copió en stdout antes de que stdout fuera redireccionada.



**&>**

La instrucción `&>` permite combinar stdout y stderr en el mismo flujo. Este nuevo flujo se puede mandar a un archivo.

```
rm archivo_no_existente &> bitacora
```

```
cat bitacora
```

```
rm archivo_existente &> bitacora
```

```
cat bitacora
```



# **3. Redirección de entrada a archivos**



**< stdin**

La entrada puede ser redirigida con el símbolo de **menor que**.

Ejemplo:

```
cat < got8.txt
```

La notación **<** es una abreviación de **<0** (recordando que stdin es el flujo 0)

```
cat <0 got.txt
```



<<

Es una manera de concatenar la entrada hasta que cierta secuencia sea encontrada(normalmente EOF).

```
cat <<EOF > texto.txt
```

La secuencia EOF se puede escribir literalmente o llamada con Ctrl+d

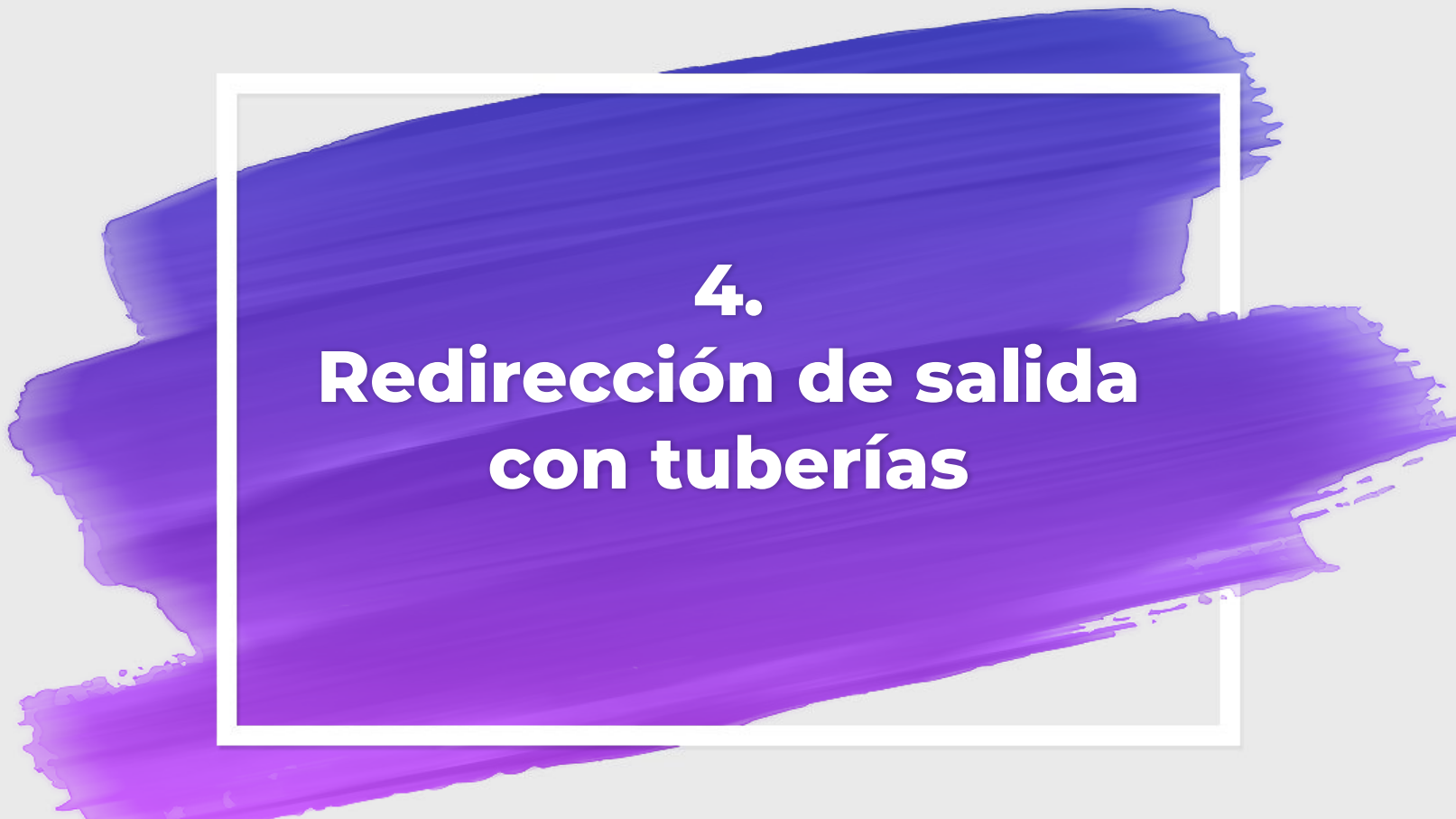




<<<

Se puede utilizar para pasar cadenas directamente a un comando. Es lo mismo que `echo <cadena> |`, pero hay un proceso menos corriendo

```
base64 -d <<< aG9sYQo=
```



# **4. Redirección de salida con tuberías**



|

Se utiliza para que la salida estándar (stdout) de un comando sea la entrada estándar (stdin) de otro comando.

Ejemplo:

```
echo $HOME | ls -l
```



## **5. Filtros**



Los comandos que fueron creados para ser utilizados con pipes, se llaman filtros. Son programas pequeños y enfocados a tareas muy específicas



**tee**

Pone entrada estándar en salida estándar y al mismo tiempo en un archivo

```
cat /etc/passwd | tee copia_passwd | tac
```





# grep

Filtra líneas de texto que contienen o no una cadena

```
cat /etc/passwd | grep $USERNAME
```

```
grep $USERNAME /etc/passwd
```

Opciones:

- i no distingue mayúsculas o minúsculas

- v resultados que no contienen una cadena



# grep

También muestra líneas antes de las coincidencias

```
grep -A1
```

También muestra líneas después de las coincidencias

```
grep -B1
```

También muestra las líneas antes y después de las coincidencias

```
grep -C1
```



# cut

Puede seleccionar columnas de archivos, dependiendo de un delimitador o un número de bytes.

```
cut -d: -f1,3 /etc/passwd | tail -1
```

```
cut -c2-7 /etc/passwd | tail -1
```



**tr**

Traduce un set de caracteres a otro

```
cat prebes.txt | tr 'e' 'E'
```

```
cat script.txt | tr [a-z] [a-Z]
```

-s (squeeze)

-d

Se puede utilizar para “cifrar” archivos



**WC**

Se utiliza para contar líneas, palabras y caracteres

`wc -l`

`wc -w`

`wc -c`

A large, vibrant green brushstroke graphic that sweeps across the left side of the slide, partially enclosed by a white rectangular frame.

# **sort**

Por default ordena alfabéticamente

`sort -k (column)`

`Sort -n (numerical)`





## uniq

Permite remover duplicados de una lista ordenada.

```
cat music.txt
```

```
sort music.txt
```

```
sort music.txt | uniq
```

```
uniq -c (cuenta ocurrencias)
```



## **comm**

Compara flujos o archivos. Por default muestra tres columnas.

```
comm lista1.txt lista2.txt
```

```
comm -12 lista1.txt lista2.txt
```

Los dígitos indican las columnas que no se deben mostrar



**od**

Permite mostrar el contenido de un archivo en diferentes formatos.

Hexadecimal

`od -t x1 prueba.txt`

Octal

`od -b prueba.txt`

ASCII

`od -c prueba.txt`



# Sed Stream Editor Sed

Puede editar un flujo usando expresiones regulares

```
echo "archivo3" | sed 's/3/42'
```

```
echo hola42 hola45 | sed  
's/hola/adios/g'
```

```
cat /etc/passwd > respaldo.txt
```

```
cat respaldo.txt | sed '/miguel/d'
```

## Ejercicios

Crea una lista ordenada de todos los usuarios del sistema en `users.txt`

Lista todos los archivos en `/etc` que contengan la cadena `'conf'` en su nombre

Escribe una instrucción que sustituya los números por letras del siguiente mensaje `'H0l4 mund0 cru3l'`

De la salida del comando `ip`, mostrar sólo la dirección IP



Now you can use any emoji as an icon!

And of course it resizes without losing quality and you can change the color.

How? Follow Google instructions

<https://twitter.com/googledocs/status/730087240156643328>

