ITEDES

Educación Digital

Módulo:

Fundamentos de Ingeniería de Software

Segmento:

Elementos informáticos

Tema:

Sistemas Operativos

Alumno. Adrián E. Lucero
luceroadrian89@gmail.com

Índice de Contenidos

Introducción....2
Respuestas....3
1. Creación de Carpetas y ficheros....3
2.*Visualizar el contenido de la carpeta en forma de lista*....4
3.*Editar fichero*....4
4. Mostrar por pantalla el contenido del Fichero....4
5. Borra la Carpeta y todo su contenido....4
6. Posicionarse en la Carpeta21....4
7. Verificar el directorio de trabajo actual....5
8. Copiar la Carpeta1 y su contenido como Carpeta3....5
9. Investigación de comandos....*5*
Conclusión....16
Bibliografía....16

Introducción

En este capitulo vamos a aprender como se crea, copia y elimina los archivos y ficheros en Linux, como así también como visualizar la ubicación actual de trabajo y aprenderemos los comando básicos a utilizar.

Respuestas

1. Creación de Carpetas y ficheros

a) Crear carpetas

1. Carpeta1
2. Carpeta2
    a. Carpeta21
3. Carpeta3

PASOS

1. cd /home/alucero/Adrian/
2. mkdir Carpeta1
3. mkdir -p Carpeta2/Carpeta21
4. mkdir Carpeta3

b) Crear ficheros

- Carpeta1
    - Fichero1
- Carpeta2
    - Fichero2
    - Carpeta21
        - Fichero21
- Carpeta3
    - Fichero3

PASOS

1. cd /home/alucero/Adrian/Carpeta1
2. touch Fichero1
3. cd ../Carpeta2
4. touch Fichero2
5. cd Carpeta21
6. touch Fichero21
7. cd ../../Carpeta3
8. touch Fichero3

2. Visualizar el contenido de la carpeta en forma de lista.

PASOS

1. cd /home/alucero/Adrian
2. ls -l

3. Editar fichero.

PASOS

1. Ubicarnos en “cd /home/alucero/Adrian/Carpeta2/Carperta21/ ”
2. Escribir “vim Fichero21” y presionar la tecla “Enter”
3. Presionar la letra “ i ” para poder insertar texto
4. Escribir “Hola soy un fichero”
5. Presionar la tecla “Esc” para guardar
6. Escribir “ :x ” y presionar la tecla “Enter” para salir de vim

4. Mostrar por pantalla el contenido del Fichero.

PASOS

1. Ubicarnos en “cd /home/alucero/Adrian/Carpeta2/Carperta21/ ”
2. Escribir “cat” y presionar la tecla “Enter”
3. Veremos que nos devuelve la consulta del archivo. En este caso, “Hola soy un fichero”

5. Borra la Carpeta y todo su contenido.

PASOS

1. Ubicarnos en “cd /home/alucero/Adrian/”
2. Escribir “rm -rf Carperta3”
3. Veremos que se borra la Carperta3 y todo su contenido por el borrado en forma recursiva.

6. Posicionarse en la Carpeta21.

PASOS

1. Para ubicarnos en la carpetas21 debemos escribir:
   cd /home/alucero/Adrian/Carpeta2/Carperta21/

7. Verificar el directorio de trabajo actual.

PASOS

1. Ubicarnos en: cd /home/alucero/Adrian/Carpeta2/Carperta21/
2. Escribir la palabra “pwd”
3. Veremos que nos devuelve el path de la ubicación actual, en este caso: “/home/alucero/Adrian/Carpeta2/Carperta21/”

8. Copiar la Carpeta1 y su contenido como Carpeta3.

PASOS

1. Ubicarnos en: cd /home/alucero/Adrian/
2. Crear la Carpeta3 escribiendo “ mkdir Carpeta3 ”
3. Ejecutar “ cp /home/alucero/Adrian/Carpeta1/* /home/alucero/Adrian/Carpeta3/ “
4. Veremos que se copio todo el contenido de la Carperta1 a la Carperta3.

9. Investigación de comandos

Administración remota

- **ssh**:

  SSH o Secure Shell, es un protocolo de administración remota que permite a los usuarios controlar y modificar sus servidores remotos a través de Internet. El servicio se creó como un reemplazo seguro para el Telnet sin cifrar y utiliza técnicas criptográficas para garantizar que todas las comunicaciones hacia y desde el servidor remoto sucedan de manera encriptada. Proporciona un mecanismo para autenticar un usuario remoto, transferir entradas desde el cliente al host y retransmitir la salida de vuelta al cliente.

Comando básico de Linux

- **clear**

  Limpia la ventana del terminal
  Con el comando clear se borran todos los comandos de la sesión.
  El usuario recibe un terminal vacío con el prompt a la espera de órdenes. Los comandos introducidos anteriormente se guardan en el scrollback buffer.
  También se puede limpiar la ventana de la consola con el atajo de teclado [CTRL] + [L]

Operación de permiso

- **chmod**

  Gestiona los permisos de acceso

  La llamada al sistema chmod (change mode) sirve para administrar los permisos en sistemas de archivos unixoides (ext2, ext3, ext4, reiser, xfs).

  La sintaxis general de chmod es:

  chmod [OPCIONES] MODO ARCHIVO

  o

  chmod [OPCIONES] MODO DIRECTORIO

  MODO corresponde a la forma de asignar los permisos que se aplica. Para saber cómo se crean y qué hay que tener en cuenta, lee nuestro artículo sobre el reparto de permisos de

  acceso con chmod.

  Con ayuda de la opción -R se pueden ampliar los derechos de forma recursiva a las subcarpetas y a los archivos contenidos en un fichero.

- **Chown**

  Administra derechos de propietario

  Normalmente, el creador de un archivo o un directorio se convierte automáticamente en su propietario (owner). La orden chown se deriva de change owner y permite configurar la propiedad de archivos y directorios.

  Este comando se utiliza basándose en la siguiente composición:

  chown [OPCIONES] [USUARIO][:[GRUPO]] ARCHIVO

  o

  chown [OPCIONES] [USUARIO][:[GRUPO]] DIRECTORIO

  Con objeto de definir derechos de propietario para usuarios o grupos se dispone de cuatro

  combinaciones posibles:

  1) El propietario y el grupo de un fichero se definen según las indicaciones:

  chown [OPCIONES] nombre_propietario:grupos_archivos.txt

  2) El grupo se modifica en función de las indicaciones pero el usuario permanece invariable:

  chown [OPCIONES] :grupos_archivos.txt

  3) Se define al propietario pero el grupo permanece invariable:

  chown [OPCIONES] propietario_archivos.txt

  4) Se vuelve a definir al propietario y el grupo se define por el grupo estándar del usuario activo:

  chown [OPCIONES] nombre_propietario: archivo.txt

  Los cambios se pueden extender a los subdirectorios con ayuda de la opción -R.

Operación de búsqueda

- **find**

  Explora el sistema de archivos

  find es un programa de líneas de comandos cuya función es buscar archivos en el sistema.

  Se utiliza en base al siguiente esquema sintáctico:

  find [OPCIONES] [RUTA] [EXPRESIÓN_DE_BÚSQUEDA][ACCIÓN]

  El directorio que se indica en RUTA define el inicio de la búsqueda, de tal forma que se exploran tanto el directorio como sus subdirectorios. Si no se indica ninguno, find comienza a explorar desde el directorio en el que se está en ese momento (directorio actual).

  Las opciones indicadas en EXPRESIÓN DE BÚSQUEDA permiten definir los criterios de búsqueda y las acciones. La acción predeterminada es -print, que muestra el resultado de la búsqueda en la salida estándar (normalmente la consola).

  Los criterios de búsqueda más habituales son el nombre del archivo (-name NOMBRE DEL ARCHIVO[SUFIJO]), un nombre de usuario (-user NOMBRE DE USUARIO), el tamaño del archivo (-size n[cwbkMG]), el momento del acceso en días (-atime [+-]n) o el momento de la modificación en días (-mtime [+-]n).

  En la búsqueda de nombres de archivo se pueden utilizar metacaracteres (comodines) como *. Si se escriben entre comillas se evita que el shell los interprete.

  Ejemplo:

  find /tmp -name "*.odt" -mtime -3 -size +20k

  Como directorio inicial se ha definido /tmp. El comando find entrega en la salida estándar todos aquellos archivos que terminan en .odt, pesan más de 20 k y fueron modificados por última vez hace menos de tres días.

  En la página del manual dedicada al comando de búsqueda find encuentras opciones adicionales.

- **Grep**

  Explora archivos de texto

  Con la orden grep (global regular expression print) se pueden explorar archivos de texto tales como archivos de registro. Como patrón de búsqueda se pueden utilizar secuencias de caracteres o expresiones regulares.

  Utiliza grep según esta sintaxis:

  grep [OPCIONES] PATRÓN_DE_BÚSQUEDA [ARCHIVOS]

  Si grep encuentra una cadena que se corresponde con el patrón dado, el terminal muestra el número de línea y especifica el nombre del archivo.

  Por regla general, grep se aplica a todos los archivos del directorio de trabajo actual. La opción -r permite la búsqueda recursiva en los subdirectorios.

Información de usuario

- **su**

  Trabajar con los permisos de otro usuario

  El comando su también permite cambiar de usuario temporalmente para realizar llamadas al sistema con los permisos de otro usuario, pero a diferencia de sudo, en este caso las órdenes no se ejecutan directamente, sino que es necesario cambiar de identidad porque su no solicita la contraseña del usuario que invoca, sino la del usuario cuyos derechos se quieren suplantar. Para invocar a programas como administrador, el usuario necesita así la contraseña root del sistema. Además, al contrario que sudo, no es posible limitar a su a un conjunto de llamadas al sistema previamente definido por el administrador.

  La sintaxis general del comando obedece a este orden:

  su [OPCIONES] [USUARIO]

  Una solicitud sin nombre de usuario selecciona a root como usuario.

- **sudo**

  Ejecuta programas con los permisos de otros usuarios

  La instrucción sudo (substitute user do) puede anteponerse a una llamada al sistema para ejecutarla con los derechos de un usuario diferente de forma segura. Generalmente se requiere la contraseña del usuario que realiza la invocación.

  Cuando se introduce el comando sin indicar ningún nombre de usuario, se utiliza el superusuario root como usuario.

  sudo LLAMADA

  En el archivo /etc/sudoers, los administradores tienen la posibilidad de definir quién puede utilizar sudo y qué llamadas se permiten. Para poder utilizar el comando sudo, el usuario ha de pertenecer al grupo sudo.

  Para seleccionar un usuario diferente se utiliza sudo con la opción -u y el nombre de usuario deseado.

  sudo -u USUARIO LLAMADA

  Un cambio de usuario así solo es posible cuando está autorizado en /etc/sudoers.

  Si de lo que se trata es de trasladarse definitivamente al shell root para ejecutar comandos con derechos de administrador se utiliza sudo con la opción -i.

  sudo -i

  El comando sudo es útil porque permite a los usuarios ejecutar comandos previamente definidos como usuarios raíz sin que sea necesario revelar la contraseña root.

Información de sistema

- **free**

Muestra la carga de la memoria RAM

El comando free muestra la ocupación de la memoria con esta sintaxis:
free [OPCIONES]
Como salida obtienes dos conceptos: Mem (Memory) y Swap.
Mem consiste en la memoria física del sistema. Si está agotada, Linux desplaza porciones de los datos almacenados en la RAM hacia el disco duro. En este caso se habla de Swap-Space.
free también soporta la opción –h, con la que obtienes la información en un formato legible.

Información red

- **ifconfig**

El comando ifconfig es utilizado para arrancar la interface de la tarjeta de red, pararla y realizar todas las configuraciones sobre dicha tarjeta en nuestros servidores dedicados o servidores virtuales.
Para detener una interfaz, por ejemplo utilizaremos el comando ifconfig <interfaz> down de la siguiente forma:
ifconfig venet0:1 down
Para volver a arrancar una interfaz, utilizaremos lo anterior pero modificando down por up.
ifconfig venet0:1 up
Otros comandos de configuración de la potente herramienta ifconfig son como por ejemplo
añadirle una máscara de red a una interfaz, por ejemplo:
ifconfig venet0:1 192.168.100.58 netmask 255.255.255.0

Gestión de proceso

- **Top**

Listado dinámico de los procesos en ejecución
Con el comando top obtienes una lista dinámica de todos los procesos activos. Para ello utiliza este esquema:
top [OPCIONES]
La salida de la información sobre los procesos se puede ajustar con ayuda de diversas opciones. El comando soporta, además, el uso de las siguientes teclas para ordenar la salida:

[P] = ordena la salida en función del uso de CPU
[M] = ordena la salida según el uso de memoria
[N] = ordena la salida numéricamente por PID
[A] = ordena la salida por edad
[T] = ordena la salida por tiempo utilizado de CPU
[U USUARIO o UID] = filtra la salida por el usuario

Utiliza la tecla [H] para acudir a las páginas de ayuda y [Q] para abandonar la vista de los procesos.

Operación en el directorio

- **ls**

  Muestra el contenido del directorio como una lista

  La orden ls equivale a list y se utiliza para mostrar el contenido de un fichero (los nombres de todos sus archivos y carpetas).

  Esta es la sintaxis de la orden:

  ls [OPCIONES] DIRECTORIO

  Si no se añade a ls ningún directorio, el comando enumera el contenido del directorio en curso.

  Con ayuda de diferentes opciones se puede definir qué información se ha de mostrar y cómo.

- **Cd**

  Navega por el árbol de ficheros

  El comando cd es la abreviatura de change directory y se utiliza para navegar por el directorio.

  La sintaxis de esta orden sigue el esquema:

  cd [OPCIÓN] DIRECTORIO

  Si no se indica ningún directorio concreto, cd cambia automáticamente al directorio principal del usuario.

  Si se acompaña de un guion (-), cd abre el directorio precedente.

- **Pwd**

  Muestra el nombre del directorio

  Con pwd (abreviatura de print working directory) la consola muestra el nombre del directorio de trabajo (en curso).

  La sintaxis del comando es: pwd [OPCIONES]

- **mkdir**

  Crea un directorio

  El comando mkdir corresponde a make directory y permite a los usuarios de Linux crear directorios desde cero.

  Para crear un directorio en el fichero en curso escribe la siguiente sintaxis:

  mkdir [OPCIÓN] DIRECTORIO

  Si lo que se necesita es crear varios ficheros a la vez, se escribe uno detrás de otro sin signos de puntuación y con espacio intermedio:

  mkdir [OPCIÓN] DIRECTORIO1 DIRECTORIO2

  Si se quiere crear un directorio nuevo en otro fichero diferente al actual se ha de indicar laruta absoluta o relativa al fichero:

  mkdir /home/user/Desktop/test

  mkdir/Desktop/test

  En ambos ejemplos se crea el directorio test en el directorio Desktop.

- **rmdir**

  Borra el directorio

  Si hiciera falta borrar un determinado directorio, se utiliza el comando rmdir (remove directory) según la siguiente sintaxis:

  rmdir [OPCIÓN] DIRECTORIO

  Con rmdir, sin embargo, solo se pueden borrar directorios vacíos. Si se quiere eliminar un fichero con todas sus carpetas y subcarpetas se utiliza el comando rm (remove) con la opción -r.

  Cuidado: rmdir no solicita confirmar el borrado. Los directorios seleccionados con el comando se borran definitivamente.

Operación en archivo

- **cp**

  Copia archivos o directorios

  La orden cp (de copy) se utiliza para copiar archivos o ficheros y sigue la siguiente sintaxis:

  cp [OPCIONES] ORIGEN DESTINO

  El elemento ORIGEN es el que se ha de copiar y DESTINO se define a un archivo o un directorio donde se ha de alojar el elemento copiado. Si se define como destino a un archivo que ya existe, el archivo origen reescribe su contenido. También se puede crear un archivo de destino nuevo. Si se han de copiar varios archivos, entonces el destino ha de ser un directorio, del mismo modo que si se copia un directorio.

  Copiar un archivo de origen en un archivo de destino en el directorio actual:

  cp [OPCIONES] ORIGEN DESTINO

  Ejemplo: cp archivo.txt archivo_copia.txt

  Copiar un archivo del directorio actual en un directorio de destino:

  cp [OPCIONES] ARCHIVO_ORIGEN DIRECTORIO_DESTINO

  Ejemplo: cp archivo.txt home/user/documentos/2017

  Copiar varios archivos en un directorio:

  cp [OPCIONES] ARCHIVO_ORIGEN1 ARCHIVO_ORIGEN2 DIRECTORIO_DESTINO

  Ejemplo: cp archivo.txt archivo.odt home/user/documentos/2017

  Copiar un directorio desde el directorio actual en otro directorio diferente:

  cp DIRECTORIO_ORIGEN DIRECTORIO_DESTINO

  Ejemplo: cp directorio1 home/user/documentos/2017

  Si se tienen que copiar todos los directorios con el contenido completo se deben incluir en el copiado todos los subdirectorios con la opción -R.

- **rm**

  Borra archivo o directorio

  El programa rm (remove) borra archivos o directorios de forma irreversible. Para ello se sigue la siguiente estructura:

  rm [OPCIONES] ARCHIVO

  o

  rm [OPCIONES] DIRECTORIO

  Para eliminar un directorio junto a todos sus subdirectorios se utiliza rm con la opción -R (-recursive):

  rm -R DIRECTORIO

  Cuando se ordena eliminar varios archivos o ficheros, estos se separan por espacios:

  rm [OPCIONES] ARCHIVO1 ARCHIVO2

- **cat**

  Agrupa contenido de varios archivos

  El comando cat (de concatenate) nace como herramienta para enlazar archivos y puede emplearse como pager para mostrar el contenido de los archivos en el terminal.

  Escribe cat con la siguiente sintaxis para leer y mostrar un archivo en la salida estándar:

  cat OPCIONES ARCHIVOS

  Si incluyes varios archivos, sepáralos mediantes espacios:

  cat OPCIONES ARCHIVO1 ARCHIVO2

  Para concatenar el contenido de varios archivos se utilizan los operadores de redirección >, < y |. Si utilizas el operador “mayor que“ (>), se aúna el contenido de dos archivos en untercero:

  cat archivo_1.txt archivo_2.txt > archivo_3.txt

- **touch**

  Cambia fecha y hora

  Para modificar la fecha y la hora de los accesos o los cambios que han tenido lugar en un archivo se utiliza el comando touch. Si se utiliza en un archivo que no existe, este se crea automáticamente, lo que hace que este comando también se utilice para generar archivos vacíos.

  Para invocar a touch se tiene que seguir este esquema:

  touch [OPCIONES] ARCHIVO

  Si de lo que se trata es de cambiar la fecha y la hora de un archivo por una fecha determinada se utiliza la opción -t incluyendo la fecha en el formato [AA]MMDDhhmm[.ss].

  Ejemplo:

  touch -t 1703231037 archivo.txt

  La fecha y la hora del acceso y los cambios respecto a este archivo se han fijado en el 23 de marzo de 2017 a las 10:37 horas. Si se prefiere, el cambio se puede limitar a la fecha de acceso o de modificación con las opciones -a y -m. Si se utiliza sin la opción -t se sella automáticamente con la fecha y la

hora actual.

Editor

- **vim**

  El editor de textos Vim

  En el caso de Vim (Vi Improved) se trata de un desarrollo ulterior del editor Vi al que supera con numerosas extensiones, como el resaltado de sintaxis, un amplio sistema de ayuda, scripting nativo, autocompletado de código y una selección de texto visual.

  Este programa de código abierto ofrece diferentes modos operativos para la edición de archivos meramente de texto y se puede utilizar tanto en el terminal como en una aplicación portátil con interfaz gráfica de usuario (GVim). Su campo central de aplicación es la edición de código de programa.

  Iniciado en la consola, Vim se utiliza con el teclado. Normalmente se invoca al comando con un archivo de texto siguiendo este esquema:

  vim [OPCIONES] ARCHIVO

  Vim guarda los archivos abiertos en una memoria caché (buffer), donde también se almacenan todos los cambios efectuados en el archivo abierto. Si se inicia Vim sin indicarle ningún archivo, el programa arranca con un buffer en blanco. Las modificaciones se aplican al archivo original solo cuando se confirma su almacenamiento con las teclas correspondientes. Si no existe ningún archivo que coincida con el indicado en el comando, se crea uno nuevo durante el almacenamiento.

  El programa vimtutor, que también se inicia en la línea de comandos, ofrece una introducción exhaustiva en Vim.

  Nuestro artículo sobre el editor Vim sirve también para profundizar en la instalación del programa y en sus diferentes modos operativos.

Pager

- **tail**

  Muestra las últimas líneas de un archivo

  Mientras que head muestra por defecto las 10 primeras líneas de un archivo, tail muestra las 10 últimas. Ambos pager se utilizan con el mismo esquema (ver head).

Archivar y Comprimir

- **tar**

  Escribe y extrae archivos en el fichero Tar
  El comando tar corresponde a tape archiver, un programa desarrollado en sus orígenes para grabar datos en unidades de cinta magnética y que hoy es uno de los programas más populares para archivar datos en Linux.
  El programa permite escribir ficheros y directorios de forma secuencial en un archivo tar y recuperarlos a partir de él. A diferencia de lo que ocurre con el formato Zip propio de sistemas Windows, con tar los derechos de usuario de los ficheros guardados no se pierden tampoco después de descomprimirlos.
  El programa de líneas de comandos tar se invoca según la siguiente sintaxis:
  tar [OPCIONES] ARCHIVOS
  Si quieres crear un archivo nuevo, utiliza tar con las opciones -c (crear nuevo archivo) y -f
  (escribir un archivo en el fichero indicado o extraerlo de él).
  En el siguiente ejemplo los archivos archivo1.txt y archivo2.txt se escriben en el recién creado fichero ejemplo.tar:
  tar -cf ejemplo.tar archivo1.txt archivo2.txt
  Si lo que quieres es que el terminal te muestre el contenido de un archivo utiliza, tar con las opciones -t (mostrar contenido de un archivo), -v (salida detallada) y -f (ver arriba):
  tar -tvf ejemplo.tar
  En caso de que los archivos empaquetados se tengan que desempaquetar en la carpeta actual, tar se acompaña entonces de las opciones -x (extraer ficheros de un archivo) y -f (ver arriba).
  tar -xf ejemplo.tar
  Con -j (bzip2), -J (xz), -z (gzip) y -Z (compress), tar también ofrece opciones que permiten
  comprimir o descomprimir archivos durante el proceso de empaquetado o extracción invocando a otro programa.
  En el ejemplo que sigue los archivos archivo1.txt y archivo2.txt se archivan en
  ejemplo.tar.gz y se comprimen con gzip.
  tar -czf ejemplo.tar.gz archivo_1.txt archivo_2.txt
  El siguiente comando extrae y descomprime todos los archivos guardados en
  ejemplo.tar.gz:
  tar -xzf ejemplo.tar.gz

Administración de procesos

- **Bg**

  El comando bg se utiliza para mover un trabajo a segundo plano. El usuario puede ejecutar un proceso en segundo plano añadiendo un símbolo "&" al final del comando.
  SINTAXIS:
  La sintaxis es
  bg [opciones] [proceso]
  OPCIONES:
  -l Informa del identificador del grupo de proceso y la carpeta de trabajo de las
  operaciones.
  -p Informa únicamente del identificador del grupo de proceso de las operaciones.
  -x Sustituye cualquier job_id encontrado en el comando o argumentos con el identificador de grupo de proceso correspondiente, después ejecuta el comando dándole argumentos.
  job Especifica el proceso que quiere ejecutarse en segundo plano.

- **fg**

  El comando fg se utiliza para situar un proceso en primer plano.
  SINTAXIS:
  La sintaxis es
  fg [especifica proceso]

- **Jobs**

  El comando jobs se utiliza para listar procesos que estés ejecutando en segundo plano o en primer plano. Si la respuesta se devuelve sin información es que no hay procesos presentes.
  SINTAXIS:
  La sintaxis es
  jobs [opciones]
  OPCIONES:
  -l Informa del identificador del grupo de proceso y la carpeta de trabajo de las operaciones.
  -n Muestra sólo los trabajos que se han detenido o cerrado desde la última notificación.
  -p Muestra sólo el identificador de proceso para los líderes de grupo de procesos delos trabajos seleccionados.
  EJEMPLO:
  jobs -l
  Muestra los trabajos que estás ejecutando en primer plano (o) en segundo plano.
  jobs -p
  Muestra sólo el identificador de proceso para los trabajos listados.

Conclusión

En conclusión hemos aprendido el manejo de los comandos básicos de Linux, pudiendo realizar las practicas de manera fácil y ágil en los distintos archivos y ficheros.

Bibliografía

- https://www.1and1.es/digitalguide/servidores/configuracion/comandos-de-linux-la-lista-fundamental/
- https://www.hostinger.com.ar/tutoriales/que-es-ssh
- https://www.hscripts.com/es/tutoriales/linux-commands/bg.html
- https://www.hscripts.com/es/tutoriales/linux-commands/fg.html
- https://www.hscripts.com/es/tutoriales/linux-commands/jobs.html
- https://blog.unelink.es/wiki/linux/el-comando-ifconfig-en-linux/