



FISI 2028

===== Introducción, git & Unix =====



Python

Universidad de los Andes
Departamento de Física



GitHub



```
: ~ > bash
```



python

C

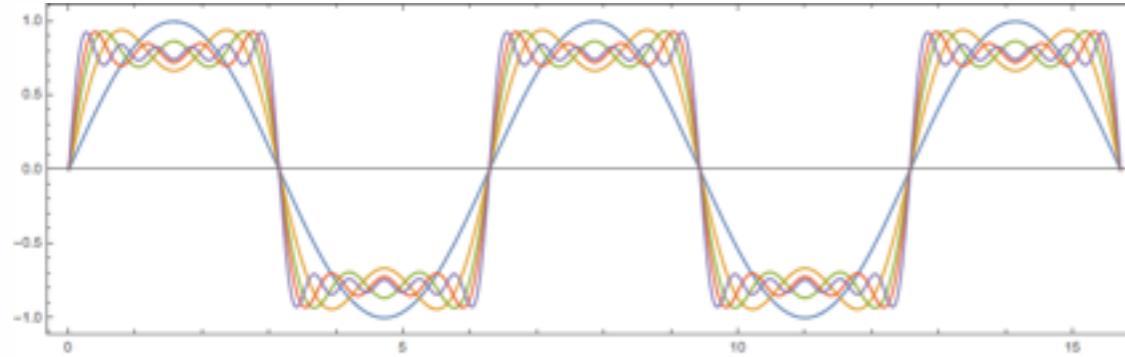


Métodos Computacionales

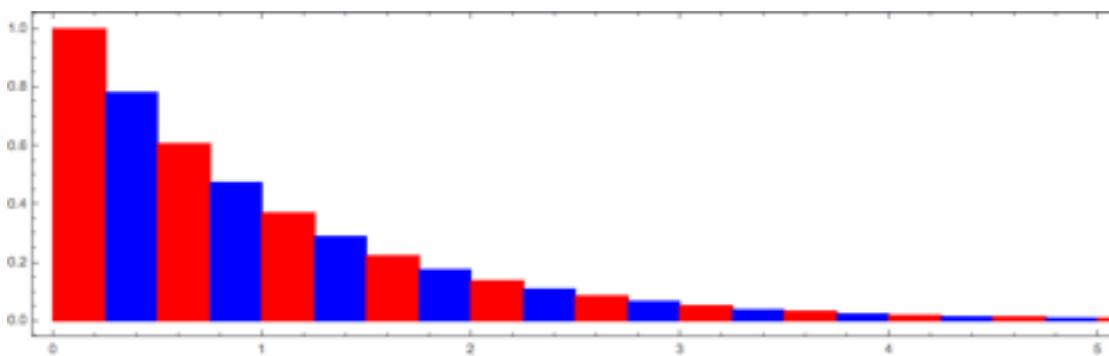
Interpolación



Análisis de Fourier



Integración y diferenciación numéricas



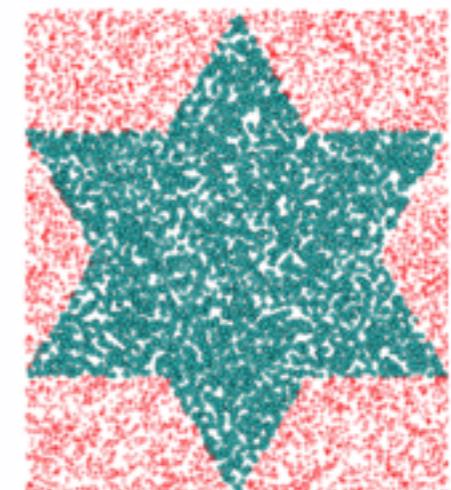
Ecuaciones diferenciales ordinarias

$$\vec{F} = m \frac{d^2 \vec{r}}{dt^2} \quad \left(-\frac{\hbar^2}{2m} \frac{d^2}{dx^2} + V(x) \right) \psi(x) = E\psi(x)$$

Ecuaciones diferenciales parciales

$$\left(-\frac{\hbar^2}{2m} \nabla^2 + V \right) \psi = i\hbar \frac{\partial \psi}{\partial t} \quad \nabla^2 \psi = \frac{1}{v^2} \frac{\partial^2}{\partial t^2} \psi$$

Monte Carlo



Referencias



Evaluación

Talleres (60%) cada semana,
exámenes (20%) cada dos semanas,
proyecto final (15%) y
Bitácora (5%).

<https://github.com/juandlizarazo/MC/LizarazoJ-Journal.md>

Todas las semanas debe haber alguna contribución a la sección **Proyecto final** y para ello vamos a reservar diez minutos de clase.

El registro de las actividades realizadas en el laboratorio es completo y detallado. La bitácora está además decorada con numerosas ``recetas'' para hacer diferentes tareas computacionales. Cada entrada va acompañada de la fecha correspondiente y no hay casi ninguna falta de ortografía o error de programación. El uso de Markdown es intachable y exhaustivo (listas, código, secciones, formato, imágenes). El registro de \textit{commits} de la bitácora muestra un trabajo constante a lo largo de todo el curso. No falta ninguna contribución semanal al hashtag \#proyectos, todas ellas son interesantes y son evidencia de gran creatividad.

Para más detalles la **rúbrica de evaluación** en el programa del laboratorio.

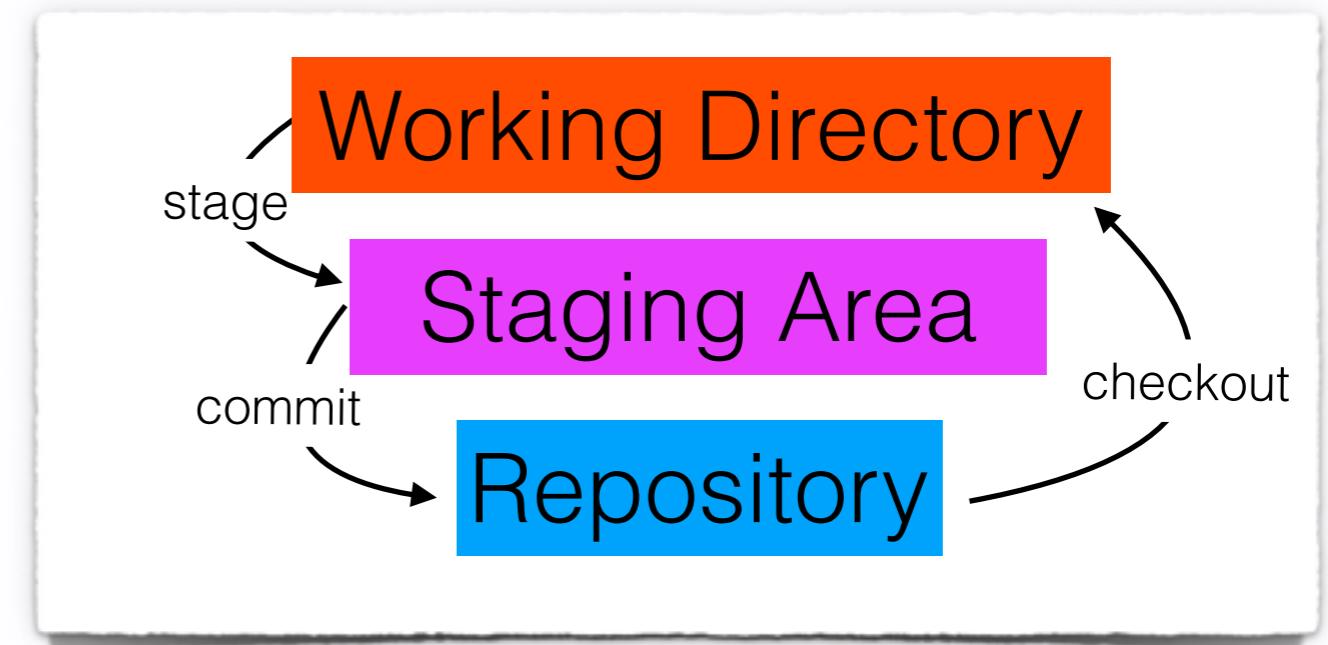
Git



```
Give all to love; // Obey thy heart; //
Friends, kindred, days, // Estate, good
fame, // Plans, credit, and the muse; //
Nothing refuse. // 'Tis a brave
master, // Let it have scope, // Follow it
utterly, // Hope beyond hope; // High and
more high, // It dives into noon, // With
wing unspent, // Untold intent; // But
'tis a god, // Knows its own path, // And
the outlets of the sky. // 'Tis not for the
mean, // It requireth courage stout, //
Souls above doubt, // Valor unbending; //
Such 'twill reward, // They shall return //
More than they were, // And ever ascending.
// // Leave all for love;- // Yet, hear
me, yet, // One word more thy heart behoved,
// One pulse more of firm endeavor, // Keep
thee to-day, // To-morrow, for ever, //
Free as an Arab // Of thy beloved. //
Cling with life to the maid; // But when the
surprise, // Vague shadow of surmise, //
Flits across her bosom young // Of a joy
apart from thee, // Free be she, fancy-free,
// Do not thou detain a hem, // Nor the
palest rose she flung // From her summer
diadem. // // Though thou loved her as
thyself, // As a self of purer clay, //
Tho' her parting dims the day, // Stealing
grace from all alive, // Heartily know, //
When half-gods go, // The gods arrive.
```

2abf3dd9643...

git es un sistema de control de versiones (VCS)
distribuido donde el estado de los datos es guardado
como una secuencia de *snapshots*



: ~> git init █

: ~> git clone █

: ~> git config █

: ~> git add █

: ~> git commit █

: ~> git pull █

: ~> git push █

: ~> git status █

Leer el primer capítulo de **Pro Git**.

Markdown



"Markdown is a markup language with plain text formatting syntax designed so that it can be converted to HTML and many other formats using a tool by the same name. Markdown is often used to format readme files, for writing messages in online discussion forums, and to create rich text using a plain text editor."

La Wikipedia

Headers

H1

H2

...

Format

italics

bold

~~scratch~~

Lists

1. First item
2. Second item
-
- * One item
- * Another item

Images

![alt-text][theimg]
[theimg]: the_url

Code

'code'

'''python
block of code
'''



Markdown Edit



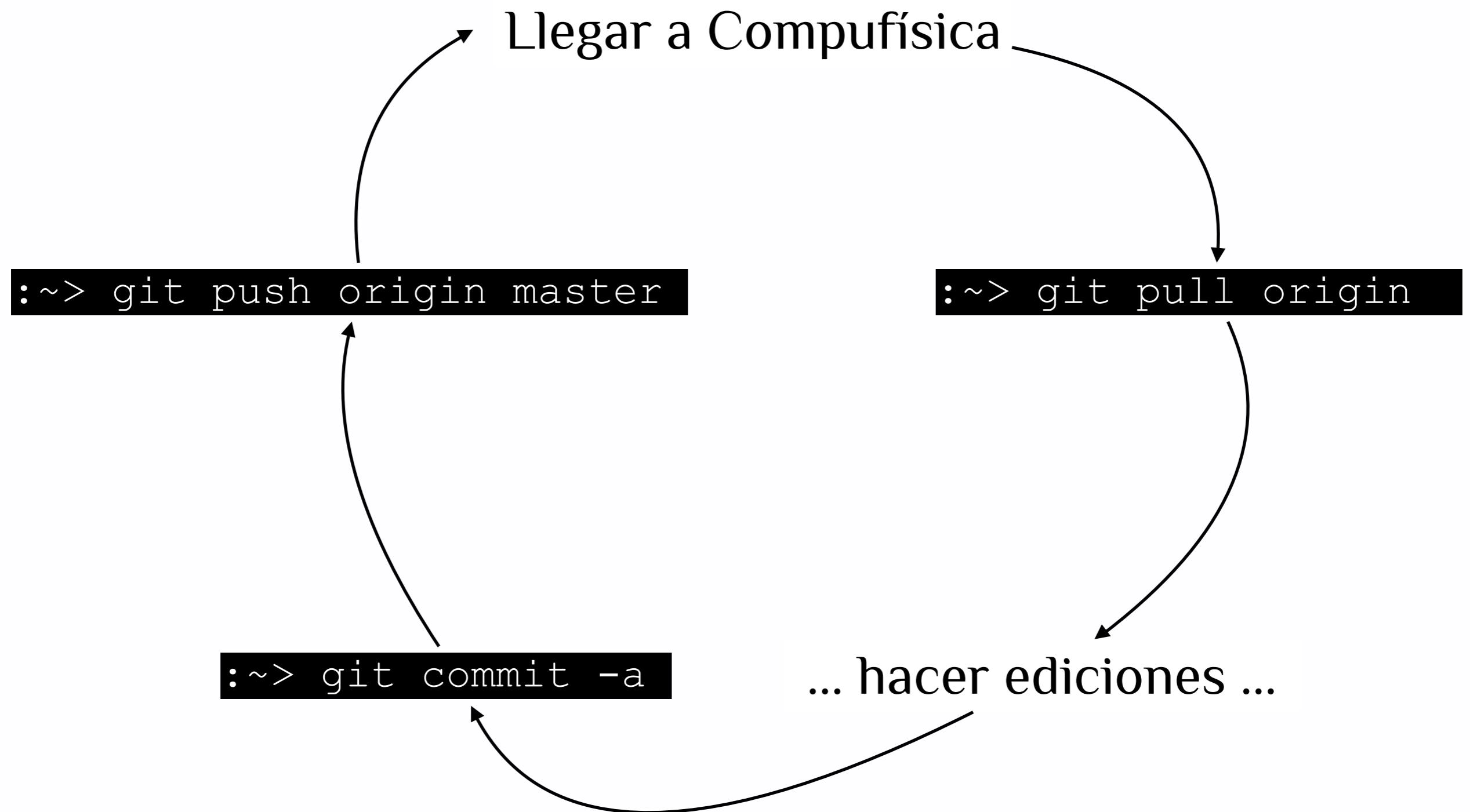
MacDown

Markdown editors.viewers

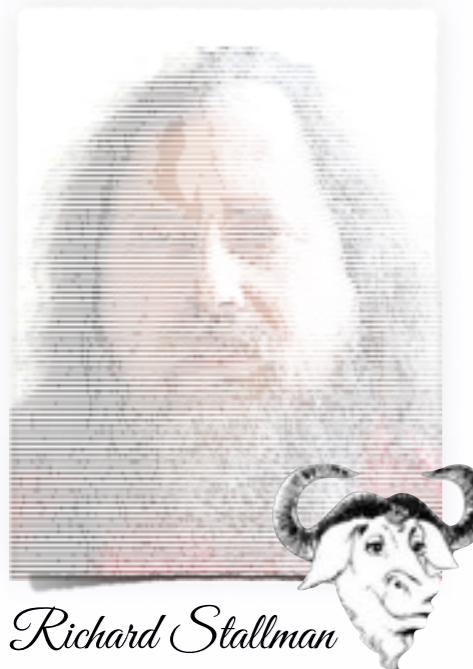
MORE

Tables, HTML, rule,
line break, YouTube

Workflow sencillo

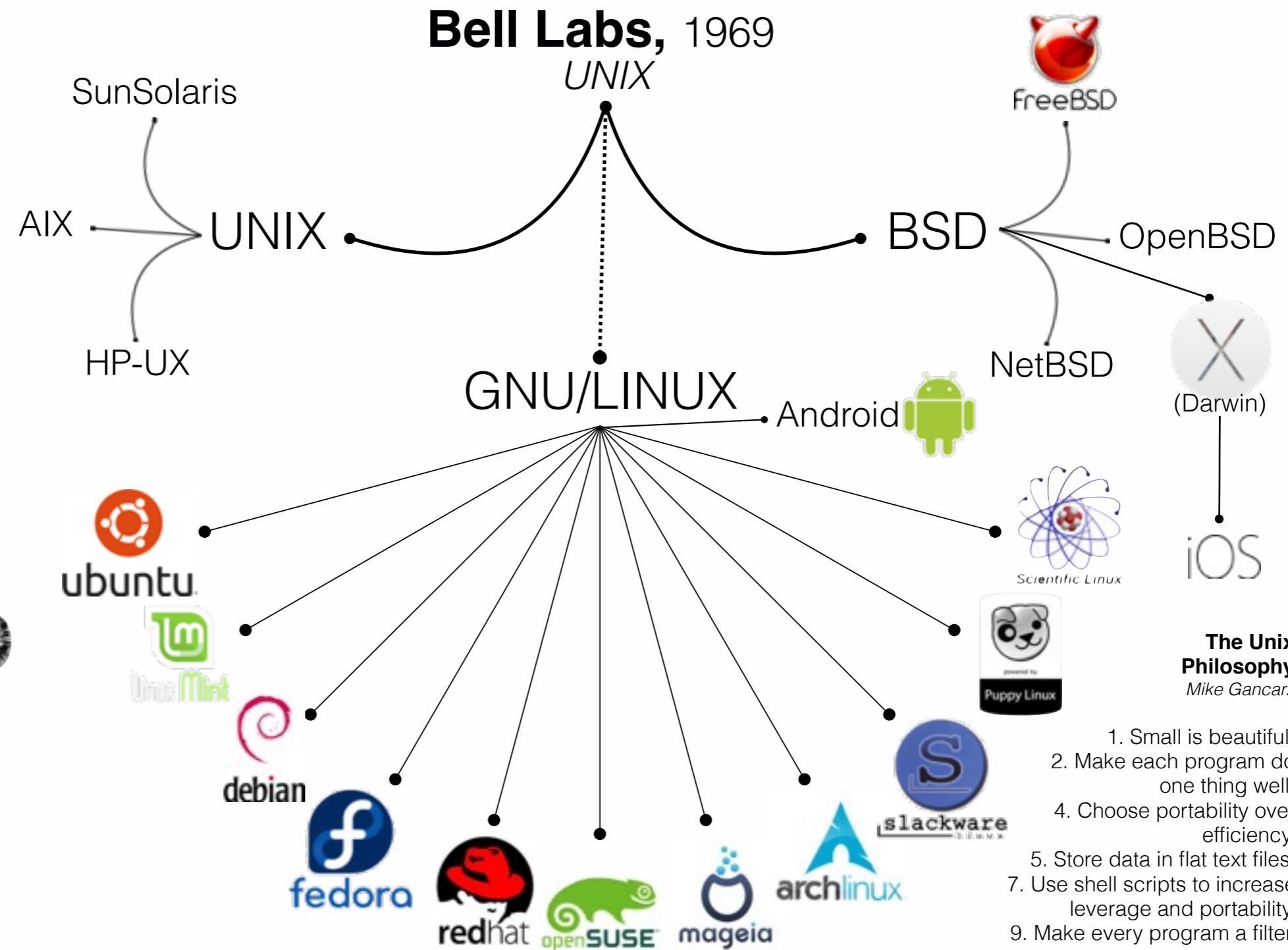


UNIX & LINUX



AT&T
Ken Thompson, Dennis Ritchie, Douglas McIlroy

Multiusuario + Multitarea + Sistema de Archivos Unificado
+ Diseño Modular + Línea de Comando



Anatomía de la terminal

```
juan — juan@Chimera: ~ — bash — 106x31
juan@Chimera:~/bin juan@Chimera:~/Scripts juan@Chimera:~/Scripts juan@Chimera:~/Scripts juan@Chimera:~ juan@Chimera:~
```

The terminal window displays a large ASCII art logo of a dragon or mythical creature, followed by a command-line interface. The command `ls` is run, listing various directories and files in the current directory.

```
juan@Chimera:~ $ ls
Applications           Dropbox          Logs            Scripts
Backups                EntropyTest      Movies          VirtualBox VMs
Biblioteca             Feynman Esalen Lectures  Music
DOS Games               Games           Otros           Wiki
Desktop                 Google Drive     Pictures        anaconda
Documents              Herr            Programming
Downloads              Library          Public          git
juan@Chimera:~ $
```

j-lizara@compufi7:~/Documents>

usuario@máquina:directorio>

cursor

En la terminal solamente se puede escribir en la posición del cursor, y una vez se ha escrito el comando deseado se ejecuta con la tecla **enter ↴**

general sintaxis: **command** option(s) filename(s)

awk

"AWK is an interpreted programming language designed for text processing and typically used as a data extraction and reporting tool. It is a standard feature of most Unix-like operating systems ... The AWK language is a data-driven scripting language consisting of a set of actions to be taken against streams of textual data – either run directly on files or used as part of a pipeline – for purposes of extracting or transforming text, such as producing formatted reports."

Imprimir la n-ésima columna (\$0 imprime toda la línea)

```
awk ' {print $n} ' file
```

Cambiar el delimitador a coma

```
awk -F", " ...
```

Hacer aritmética entre columnas

```
awk ' {print $n + $m} ' file
```

Actuar de acuerdo a condicionales: OR -> ||, AND -> &&

```
awk ' {if ($1 == 2) print $1} ' file
```

Pasar una variable de bash a awk.

```
awk -v varawk=$varbash
```

sed

"sed (stream editor) is a Unix utility that parses and transforms text, using a simple, compact programming language. sed was developed from 1973 to 1974 by Lee E. McMahon of Bell Labs, and is available today for most operating systems. sed was based on the scripting features of the interactive editor ed ("editor", 1971) and the earlier qed ("quick editor", 1965–66). sed was one of the earliest tools to support regular expressions, and remains in use for text processing, most notably with the substitution command."

Reemplazar todas las ocurrencias de *a* por *b*.

sed 's/a/b/g'

curl

"cURL (/kə:(r)l/) is a computer software project providing a library and command-line tool for transferring data using various protocols. The cURL project produces two products, libcurl and cURL. It was first released in 1997. The name is a recursive acronym that stands for Curl URL Request Library."

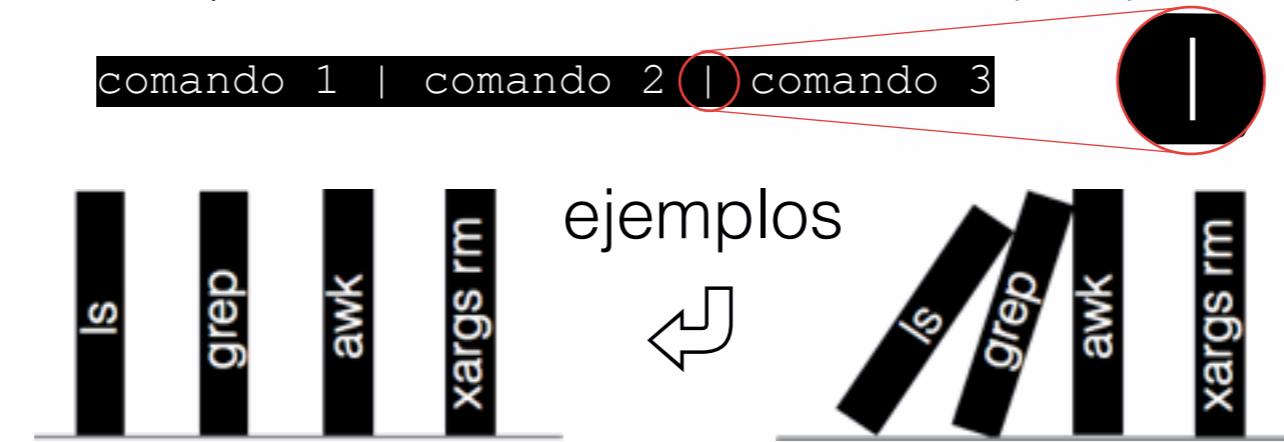
Obtener el código fuente de una url

curl url

Pipes

#c|oo\$c#-#c-|#c@c|@#oo#oo÷÷“÷“oo÷¬“oo\$c#oo|@oo\$coo÷#oo\$c|c#oo|#@c|@#oo|@c-|#coo|-÷÷“¬÷oo@#oo¬#c|oo#@oo\$c#oo|@c oo|#@oo|#c oo|@c#oo|#@oo|c oo||@#c oo#c oo\$c@÷#oo¬“oo#

Por separado cada comando es útil, y lo son más cuando la salida de uno (stdout) se puede alimentar a la entrada de otro (stdin).



```
ls -lh *.pdf | grep archivo | awk '{print $9}' | xargs -0 rm
```

>>>>> Redirección >>>>>

#c|oo\$c#-#c-|#c@c|@#oo#oo÷÷“÷“oo÷¬“oo\$c#oo|@oo\$coo÷#oo\$c|c#oo|#@c|@#oo|@c-|#coo|-÷÷“¬÷oo@#oo¬#c|oo#@oo\$c#oo|@c oo|#@oo|#c oo|@c#oo|#@oo|c oo||@#c oo#c oo\$c@÷#oo¬“oo#

La salida por defecto (stdout) es la pantalla, pero en ocasiones es de utilidad **redirigir** la salida a un archivo, esto se logra de la siguiente forma:

comando > file redirige la salida del comando al archivo *file*, el archivo *file* queda **solo** con el contenido enviado por el comando,

comando >> file redirige la salida del comando y la **añade** a *file*, el archivo *file* conserva su contenido original con la salida del comando añadida al final.

Estructura del Sistema de Archivos

Permisos y Dueños

Cada archivo pertenece a un usuario y a un grupo, y además de tener permisos para ellos dos, y para otros: user, group, others.

Los permisos son de lectura (**r**), escritura (**w**), y ejecución (**x**). Los grupos son conjuntos de usuarios.

Si se ejecuta el comando *ls -l* se listan los archivos junto con sus permisos en la primera columna

-	r W X	r - -	r - -
	user	group	others

↑
tipo
de archivo

El usuario dueño de este archivo puede leer, escribir y ejecutar; el grupo y el resto de usuarios solo pueden leer. El archivo es un archivo regular (-)

d	r W X	r W -	- - -
	user	group	others

↑
tipo
de archivo

El usuario dueño de este archivo puede leer, escribir y ejecutar; el grupo puede leer y escribir, mas no ejecutar, el resto de usuarios no tienen ningún permiso. El archivo es un directorio (d).

Contiene comandos esenciales como ls y cp
Archivos de configuración globales
Unidades de almacenamiento removibles
Directorios de los usuarios

/bin
/etc
/media
/usuarios
—/usuario1
—/usuario2
—/usuario3
—/Documents
—/Downloads
—/...
...

cd

cambia el directorio a home

cd ..

desciende un nivel en la estructura de directorios

cd dirname

moverse al directorio dirname

groups

lista los grupos a los que pertenece el usuario

chown u+x file

añade (+) permiso de ejecución (x) al archivo *file* para el usuario (u)

~

alias para el home del usuario

..

alias para el directorio padre del actual

Estructura de archivos típica

ssh & sftp

"Secure Shell, or SSH, is a cryptographic (encrypted) network protocol for initiating text-based shell sessions[clarification needed] on remote machines in a secure way."

```
: ~> ssh usuario@machine
```

```
: ~> ssh j-lizara@compufi[1-25].uniandes.edu.co
```

```
: ~> exit
```

"In computing, the SSH File Transfer Protocol (also Secure File Transfer Protocol, or SFTP) is a network protocol that provides file access, file transfer, and file management functionalities over any reliable data stream."

```
: ~> sftp usuario@machine
```

```
: ~> get remotefile
```

```
: ~> put localfile
```

```
: ~> cd, ls, help
```



PuTTY, PSFTP



Terminal

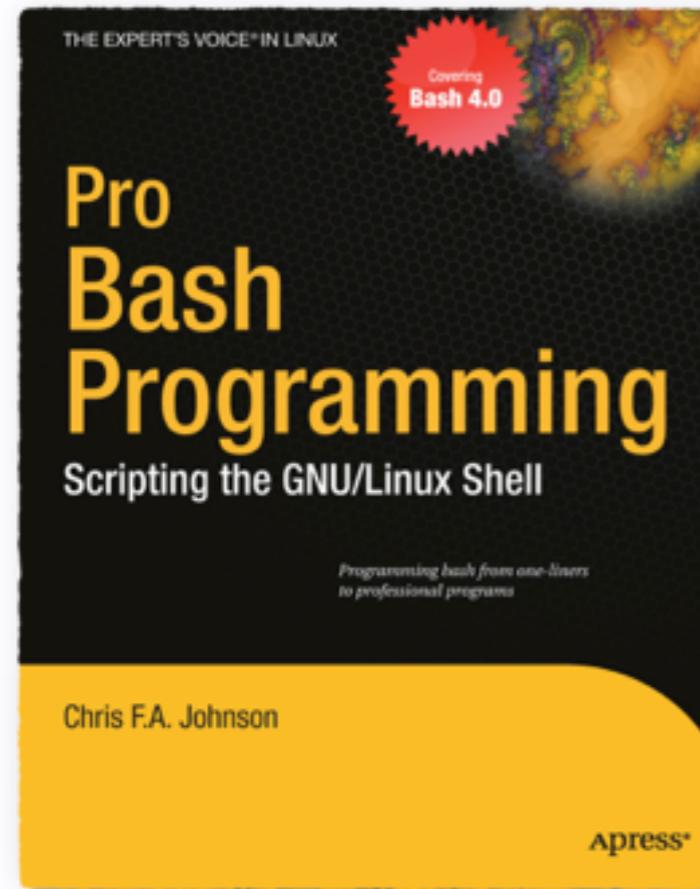
Scripts

Ventajas del Lenguaje **Shell**

Interfaz sencilla y sin costuras con cientos de utilidades Unix.

Expansión automática de comodines en listas de nombres de archivo.

Listas contenidas en una variable son automáticamente separadas en sus partes constituyentes.



Otra referencia



first line references the interpreter to be used.

Everything after **#**, and until the end of the line, is ignored.

Para poder ejecutar el script este debe tener privilegio de ejecución

```
chmod +x script.sh  
./script.sh
```

```
#!/bin/bash
# Date          : 04-Aug-2011
# Author        : Juan David
# Description   : This is my first script
echo "Hello World!" > #This prints the message
message="This is another message"
echo "$message"
# Create a new directory and give it a prefix
pref="toca"
mkdir $pref
cd toca
for i in {1..100}
do
    touch "$pref-$i"
done
# List the files
ls
```

Control de Procesos

top muestra información actualizada sobre los procesos en curso

```
juan_da/bim juan_Scripts ... juan_Scripts juan_Scripts juan_eric ~ jdiliza_s11: ~
Hands-on — jdilizarazo10@postfis11: ~ ssh ~ 83x59
top - 18:31:47 up 47 days, 11:11, 2 users, load average: 0.00, 0.01, 0.05
Tasks: 206 total, 1 running, 205 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.1 us, 0.1 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 8063424 total, 3887432 used, 4175992 free, 426860 buffers
KiB Swap: 9764860 total, 0 used, 9764860 free, 2162340 cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
14488 jdilizaro 20 0 2316m 88m 17m S 0.7 1.1 12:00.13 dropbox
16934 jdilizaro 20 0 24820 1728 1164 R 0.7 0.0 0:00.13 top
  1 root 20 0 24592 2512 1372 S 0.0 0.0 0:01.72 init
  2 root 20 0 0 0 0 S 0.0 0.0 0:00.59 kthreadd
  3 root 20 0 0 0 0 S 0.0 0.0 1:14.72 ksoftirqd/0
  6 root rt 0 0 0 0 S 0.0 0.0 0:00.10 migration/0
  7 root rt 0 0 0 0 S 0.0 0.0 0:09.66 watchdog/0
  8 root rt 0 0 0 0 S 0.0 0.0 0:00.12 migration/1
 10 root 20 0 0 0 0 S 0.0 0.0 0:20.94 ksoftirqd/1
 11 root rt 0 0 0 0 S 0.0 0.0 0:08.84 watchdog/1
 12 root rt 0 0 0 0 S 0.0 0.0 0:00.10 migration/2
 14 root 20 0 0 0 0 S 0.0 0.0 0:21.86 ksoftirqd/2
 15 root rt 0 0 0 0 S 0.0 0.0 0:09.05 watchdog/2
 16 root rt 0 0 0 0 S 0.0 0.0 0:00.24 migration/3
 18 root 20 0 0 0 0 S 0.0 0.0 0:20.52 ksoftirqd/3
 19 root rt 0 0 0 0 S 0.0 0.0 0:09.01 watchdog/3
 20 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 cpuset
 21 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 khelper
 22 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kdevtmpfs
 23 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 netns
 25 root 20 0 0 0 0 S 0.0 0.0 0:24.31 sync_supers
 26 root 20 0 0 0 0 S 0.0 0.0 0:00.75 bdi-default
 27 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kintegrityd
 28 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kblockd
 29 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 ata_sff
 30 root 20 0 0 0 0 S 0.0 0.0 0:00.02 khubd
 31 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 md
 34 root 20 0 0 0 0 S 0.0 0.0 0:02.12 khungtaskd
 35 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kswapd0
 36 root 25 5 0 0 0 S 0.0 0.0 0:00.00 ksmd
 37 root 39 19 0 0 0 S 0.0 0.0 0:00.00 khugepaged
 38 root 20 0 0 0 0 S 0.0 0.0 0:00.00 fsnotify_mark
 39 root 20 0 0 0 0 S 0.0 0.0 0:00.00 ecryptfs-kthrea
 40 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 crypto
 49 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kthrotld
 53 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 binder
 72 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 deferwq
 73 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 charger_manager
 74 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 devfreq_wq
 220 root 20 0 0 0 0 S 0.0 0.0 0:00.00 scsi_eh_0
 221 root 20 0 0 0 0 S 0.0 0.0 0:00.00 scsi_eh_1
 227 root 20 0 0 0 0 S 0.0 0.0 0:00.00 scsi_eh_2
 229 root 20 0 0 0 0 S 0.0 0.0 0:00.00 scsi_eh_3
 231 root 20 0 0 0 0 S 0.0 0.0 0:00.00 scsi_eh_4
 232 root 20 0 0 0 0 S 0.0 0.0 0:00.00 scsi_eh_5
 282 root 20 0 0 0 0 S 0.0 0.0 0:07.52 jbd2/sda5-8
 283 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 ext4-dio-unwrit
 370 root 20 0 17232 636 456 S 0.0 0.0 0:00.13 upstart-udev-br
 374 root 20 0 22100 1936 832 S 0.0 0.0 0:00.06 udevd
 417 root -51 0 0 0 0 S 0.0 0.0 0:00.02 irq/46-me
 454 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kpamoused
 741 root 20 0 15188 628 424 S 0.0 0.0 0:00.05 upstart-socket-
```

Un **proceso** es un programa en ejecución, y consiste del código y datos que le corresponden. Un **trabajo** es una tarea que ha sido enviada al *background* o que ha sido sometida para su posterior ejecución desde una terminal específica.

Cuando se ejecuta un comando en la terminal esta queda inhabilitada hasta que el comando termine de ejecutarse, sin embargo, si se presiona **^c** entonces se cancela el comando, y si se presiona **^z** este se pone en pausa.

Si inmediatamente de poner en pausa un trabajo se ejecuta el comando **bg** entonces el comando se reinicia, y continua su ejecución en el *background*. Procesos que corren en el *background* continúan en ejecución después de que se cierra la terminal (usando **exit**), e incluso después de que se cierra la sesión. Una forma abreviada de enviar un proceso al *background* consiste en poner & al final del comando a ejecutar, e. g. **sleep 100 &**.

El comando **ps** lista todos los procesos que corren bajo el usuario de la terminal, el **PID** (process ID) identifica el proceso y permite cancelarlo mediante el comando **kill -9** PID.

El comando **jobs** lista los trabajos asociados a la terminal. ejemplos

vi

From Wikipedia, the free encyclopedia

For other uses, see [Vi \(disambiguation\)](#).

vi /vɪˈaɪ/ is a screen-oriented [text editor](#) originally created for the [Unix](#) operating system. The portable subset of the behavior of vi and programs based on it, and the [ex](#) editor language supported within these programs, is described by (and thus standardized by) the [Single Unix Specification](#) and [POSIX](#).^[1]

The original code for vi was written by [Bill Joy](#) in 1976, as the visual mode for a [line editor](#) called ex that Joy had written with Chuck Haley.^[2] Bill Joy's ex 1.1 was released as part of the first [BSD Unix](#) release in March, 1978. It was not until version 2.0 of ex, released as part of Second Berkeley Software Distribution in May, 1979 that the editor was installed under the name vi (which took users straight into ex's visual mode), and the name by which it is known today. Some current implementations of vi can trace their source code ancestry to Bill Joy; others are completely new, largely compatible reimplementations.

The name vi is derived from the shortest unambiguous abbreviation for the command `visual` in ex; the command in question switches the [line editor](#) ex to [visual](#) mode. The name vi is pronounced /vɪˈaɪ/^{[3][4][5]} (as in the discrete English letters v and i).

In addition to various non-free software implementations of vi distributed with proprietary implementations of Unix, several free and open source software implementations of vi exist. A 2009 survey of [Linux Journal](#) readers found that vi was the most widely used text editor among respondents, beating [gedit](#), the second most widely used editor, by nearly a factor of two (36% to 19%).^[6]

vi



WIKIPEDIA
The Free Encyclopedia



vi editing a [Hello World program in C](#). Tildes signify lines not present in the file.

Developer(s)	Bill Joy
Initial release	1976, 37–38 years ago
Written in	C
Operating system	Unix-like
Type	Text editor
License	BSD License

movimiento del cursor

h - izquierda

j - abajo

k - arriba

l - derecha

modo insertar

i - insertar después del cursor

Esc - salir del modo

salir

:w - guardar el archivo sin salir

:wq - guardar y salir

:q - salir

:q! - salir sin guardar cambios

Editores de texto



Emacs

(editores de texto en la terminal)



(editor para Windows)



TextWrangler
(editor para Mac)

The Unix
Philosophy
Mike Gancarz

1. Small is beautiful.
2. Make each program do one thing well.
4. Choose portability over efficiency.
- 5. Store data in flat text files.**
7. Use shell scripts to increase leverage and portability.
9. Make every program a filter.