

## Console Ninja: Como dejar de ser un usuario desconsolado



Universidad Tecnológica Nacional  
Nicolás Brailovsky  
27 de noviembre del 2009

- 1 Introducción
- 2 Aprendiendo a caminar
- 3 Temas menos básicos
- 4 Scripting
- 5 Permito!
- 6 Review



# Outline

1 Introducción

2 Aprendiendo a caminar

3 Temas menos básicos

4 Scripting

5 Permisos!

6 Review



# ¿Por qué?

*Hace falta?*

- Como usuarios expertos, a veces no hay otra
- Es práctico para automatizar tareas
- Está en todas las instalaciones
- Si. En serio.



# ¿Por qué?

*No, en serio. Hace falta?*

- Parece complicado... pero solo al principio!
- Se hace costumbre; memoria mecánica
- Más rápido
- Suma geek points



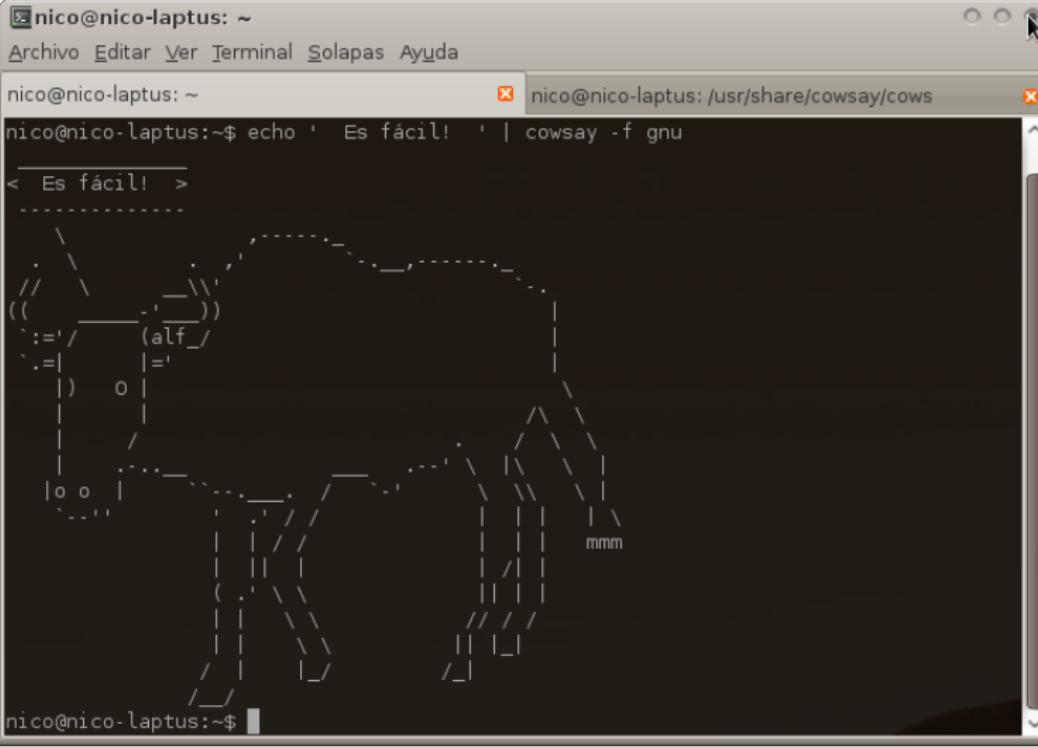
# ¿Por qué?

*No, en serio. Hace falta?*

- Parece complicado... pero solo al principio!
- Se hace costumbre; memoria mecánica
- Más rápido
- Posta



# User friendly?



A screenshot of a Linux desktop environment showing a terminal window. The terminal has two tabs:

- The active tab shows the command: `nico@nico-laptus:~$ echo ' Es fácil!' | cowsay -f gnu`. The output is a cow saying "Es fácil!".
- The other tab shows the command: `nico@nico-laptus: /usr/share/cowsay/cows`.

The cow art is rendered in ASCII characters, featuring a head labeled "alf", a body with legs, and a tail labeled "mmm". A small penguin icon is visible in the bottom right corner of the desktop.

# User friendly?

- Es intimidante
- Hay una barrera de entrada

*Pero...*

- Es una herramienta poderosa
- Imprescindible en servidores
- Mucha información online y offline
- La memoria mecánica facilita el uso
- Hasta Microsoft quiere una consola!



# User friendly?

- Es intimidante
- Hay una barrera de entrada

*Pero...*

- Es una herramienta poderosa
- Imprescindible en servidores
- Mucha información online y offline
- La memoria mecánica facilita el uso
- Hasta Microsoft quiere una consola!  
**(FAIL)**



# User friendly?



Adam Frucci-Simodo



# ¿Qué se puede hacer?



*Algunas tareas son prácticas*

- Administrar el sistema
- Editar texto, programar
- Conectarse a servidores remotos

*Otras son ridículas*

- Ver películas e imágenes
- Navegar por internet
- Escuchar música



# ¿Qué se puede hacer?

Pero lo importante: **no es voodoo!**



*Algunas tareas son prácticas*

- Administrar el sistema
- Editar texto, programar
- Conectarse a servidores remotos

*Otras son ridículas*

- Ver películas e imágenes
- Navegar por internet
- Escuchar música



# Outline

1 Introducción

2 Aprendiendo a caminar

3 Temas menos básicos

4 Scripting

5 Permito!

6 Review



# Superando el shock inicial

## *Algunos comandos básicos*

- cd
- ls
- cat
- nano
- man
- vim (OK, mentí, no es tan básico)

Recordar: importa la memoria mecánica



# Un ejemplo

*Pensemos como Homero...*

- Completamos la lista de compras
- Nos fijamos que hay
- Y compramos más cerveza



# Un ejemplo

*Pensemos como Homero...*

```
1 $ cd ~/supermercado
2 $ ls
3 lista pedido_anterior
4 $ cat lista
5 Papas fritas
6 Cerveza
7 Cerveza
8 $ echo "Mas cerveza" >> lista
```



# Tips de usabilidad



- Imprimir un cheatsheet al empezar
- tab tab tab tab
- Control + R  
(buscar comando anterior)
- tab tab tab tab



# Otro ejemplo

*Pensemos como Homero ...*

- Para La Odisea *echo* no alcanza
- Se puede usar Vim o emacs
- nano y pico son más simples



# Otro ejemplo

*Pensemos como Homero . . . (no, el otro)*

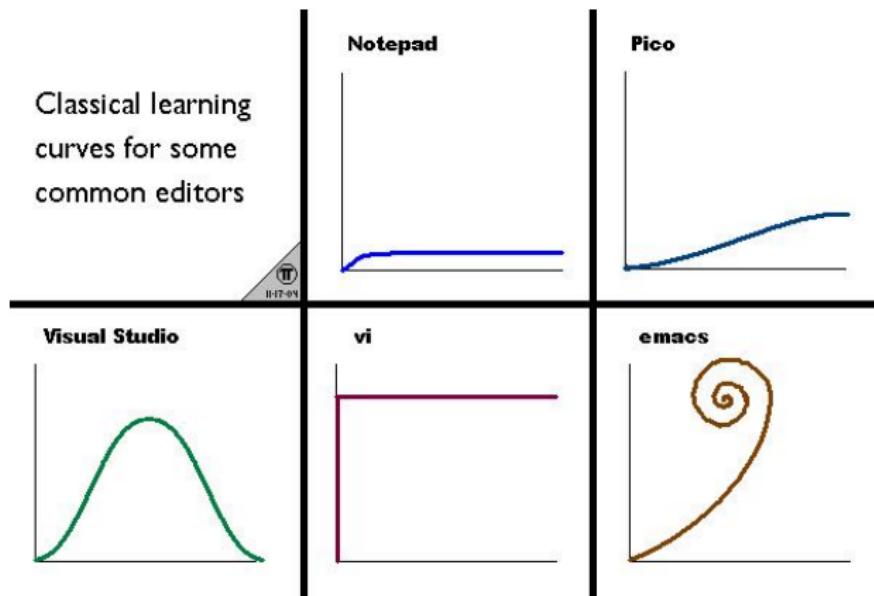
- Para La Odisea *echo* no alcanza
- Se puede usar Vim o emacs
- nano y pico son más simples



# Editores

*Hay muchos editores para elegir...*

- Vim y emacs son más geek (power!)
- No reiniciar si no entendemos el Vim (:wq)



# Outline

- 1 Introducción
- 2 Aprendiendo a caminar
- 3 Temas menos básicos
- 4 Scripting
- 5 Permisos!
- 6 Review



# Servidores remotos

*Sigamos pensando en Homero...*

- Como copiamos La Odisea de la oficina a casa?
- Usando scp:  
`scp laodisea homero@grecia:~a`
- scp no es más que cp sobre ssh
- ssh es un protocolo seguro
- ssh (el comando) ejecuta algo en un servidor



---

<sup>a</sup>Se acuerdan del ~?



# Tareas programadas

*Sigamos pensando en Homero...*

- Queremos hacer un backup
- Mañana a la mañana
- Pero nos olvidamos, todavía no se inventó el reloj



# Tareas programadas

*Sigamos pensando en Homero...*

- Queremos hacer un backup
- Mañana a la mañana
- Pero nos olvidamos, todavía no se inventó el reloj
- Fácil, usamos *at*

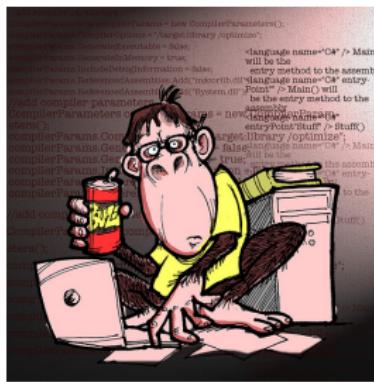
```
1 $ at tomorrow 7:00 am
2 at> cp ~/odisea /backups/$(date +"%F")
3 at> <C-D>
4 job 42 at Sat Nov 28 7:00:00 2009
5 $ at -l
6 42 Sat Nov 28 7:00:00 2009 a nico
7 $
```



# Códigos secretos

*Hay muchos secretos en el ejemplo anterior...*

- `~` => home sweet home
- `<C-D>` => Control + D
- `date +"%F"` => Mostrar fecha en formato Y-M-D
- `$(...)` => evaluar comando antes  
Ej: `echo "Hola $(echo mundo)"`



# Tareas programadas

*Sigamos pensando en Homero...*



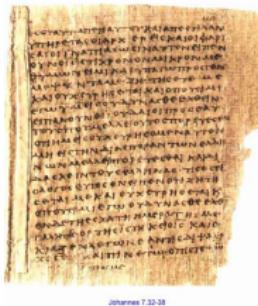
Johannes 7:32-38

- ¿Y si el backup es todos los días?



# Tareas programadas

*Sigamos pensando en Homero...*



Johannes 7:32-38

- ¿Y si el backup es todos los días?
- Fácil, usamos *cron*

- Leer *man crontab*
- Agregar el script en /etc/crontab
- Recordar que todavía no escribimos un script!
- Pasar a la siguiente sección y explicar que es un script



# Outline

- 1 Introducción
- 2 Aprendiendo a caminar
- 3 Temas menos básicos
- 4 Scripting
- 5 Permisos!
- 6 Review



# Lo qué?



- La consola permite automatizar tareas
- Una forma simple: scripting!
- Script (bash): secuencia de comandos
- No son más que texto plano
- La primera línea identifica el intérprete
- Bash es un intérprete
- También lo son zsh, ksh, PHP, Ruby...



# Un ejemplo

*Armemos un script para que Homero tenga su backup*

```
1 #!/bin/bash
2
3 echo "Copiando laodisea a grecia..."
4 scp laodisea homero    grecia:~
5 echo "Fin!"
```



# Un ejemplo

*Armemos un script para que Homero tenga su backup*

```
1 #!/bin/bash
2
3 ARCHIVO=laodisea
4 SERVIDOR_REMOTE=grecia
5 USUARIO=homero
6
7 echo "Copiando $ARCHIVO a $SERVIDOR_REMOTE..."
8 scp $ARCHIVO $USUARIO $SERVIDOR_REMOTE:~
9 echo "Fin!"
```



# Un ejemplo

*Armemos un script para que Homero tenga su backup*

```
1 #!/bin/bash
2
3 ARCHIVO=laodisea
4 SERVIDOR_REMOTE=grecia
5 USUARIO=homero
6
7 echo "Copiando $ARCHIVO a $SERVIDOR_REMOTE..."
8 # Comprimir el archivo antes de enviarlo
9 tar -xvzf $ARCHIVO.tar.gz $ARCHIVO
10 scp $ARCHIVO.tar.gz $USUARIO $SERVIDOR_REMOTE:~
11 rm $ARCHIVO.tar.gz
12 echo "Fin!"
```



# Corriendo scripts

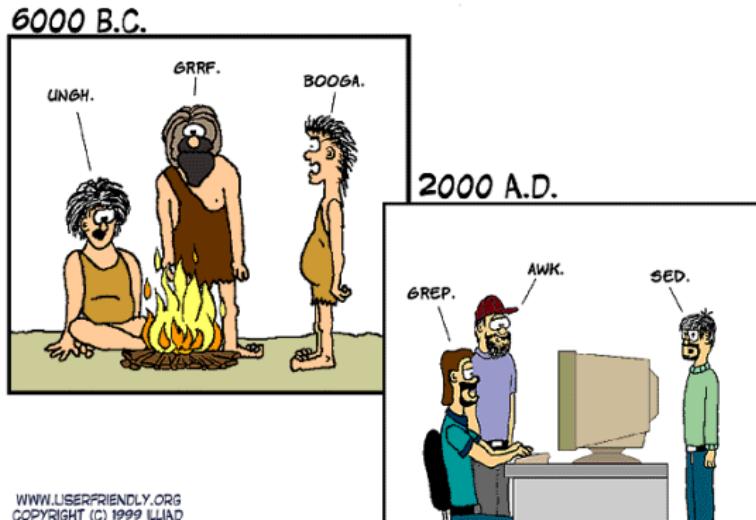
*Para recordar*

- El script se ejecuta con  
`./nombre_del_script`
- El `./` ubica el directorio actual  
¿Cuál es el resultado de `echo $PATH` ?
- No olvidarse:  
**chmod +x nombre\_del\_script**  
Es decir, darle permiso de ejecución. . .



# Scripting avanzado

- Hay scripts complejos también: script bash == programa!
- Queda como tarea (o para otra charla)
  - Pipes: redirigir la salida de un comando a otro
  - sed: editor de texto por comando (!)
  - awk: lenguaje procesador de texto
  - grep: filtros de texto



# Scripting avanzado

- Hay scripts complejos también: script bash == programa!
- Queda como tarea (o para otra charla)
  - Pipes: redirigir la salida de un comando a otro
  - sed: editor de texto por comando (!)
  - awk: lenguaje procesador de texto
  - grep: filtros de texto



# Outline

1 Introducción

2 Aprendiendo a caminar

3 Temas menos básicos

4 Scripting

5 Permito!

6 Review



# Seguridad



- En Linux hay usuarios y grupos
- Usuarios: pertenecen a grupos
- Archivos: pertenecen a grupos y usuarios
- Cada uno tiene acceso por usuario y grupo



# Woot?

*Repasemos:* Los usuarios pertenecen a grupos



# Woot?

*Repasemos:* Los archivos pertenecen a grupos y usuarios



# Modos de archivo



**Read**



**Write**



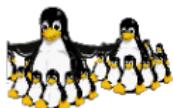
**Execute (X)**



# Modos de archivo



User



Group



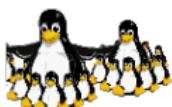
Other



# Modos de archivo: modo octal



User



Group



Other



# Modos de archivo: modo octal



User



1



1



1



Group



1



1



0



Other



1



0



0



# Modos de archivo: modo octal



User



1



1



1



Group



1



1



0



Other



1



0



0



# Seguridad



- **useradd / adduser** : crear usuario
- **/etc/group** : usuarios por grupo
- **chown** : cambiar propietario
- **chmod** : cambiar permiso



# Seguridad



- Creamos un directorio:  
`mkdir test`
- Creamos algunos archivos:  
`touch archivo`
- ¿Cómo asignamos owner y grupo?  
`chown user:group archivo`
- ¿Cómo le damos los permisos del ejemplo?  
`chmod 764 archivo`



# Outline

- 1 Introducción
- 2 Aprendiendo a caminar
- 3 Temas menos básicos
- 4 Scripting
- 5 Permisos!
- 6 Review



# Aprendimos algo?

*No lo se, pero igualmente hoy vimos...*

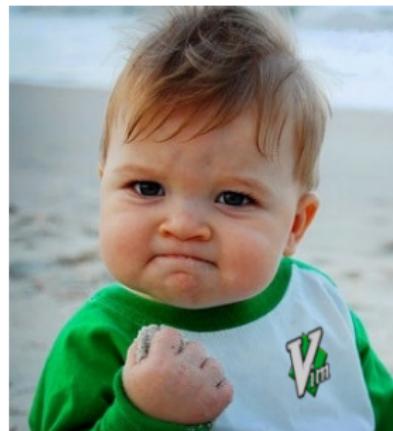


- Para qué sirve la consola
- Cómo familiarizarse con la consola
- Los comandos básicos (de la consola)
- Qué al decir Homero todos piensan en cerveza
- Cómo acceder a un servidor remoto
- Cómo automatizar y programar tareas



# Aprendimos algo?

- Hay muchos editores
- Algunos sirven para programar
- Hay que aprender a usarlos



# Podemos agregar...

*No todo son "tareas útiles"*

- cowsay
- festival
- mocp
- lynx



# Console Ninja: Como dejar de ser un usuario desconsolado

## Gracias por venir!



Nicolás Brailovsky  
con



nicolas.brailovsky@icloud.com http://www.nicolas-brailovsky.com

<http://www.gnutn.org.ar>  
<http://nicolasb.com.ar>