

Introducción a Bash Scripting

Clase 3 - 7/3/2024



Bootcamp de DevOps @ Código Facilito



Alejandra Bricio



cloud, scripting, k8s
@alebricio

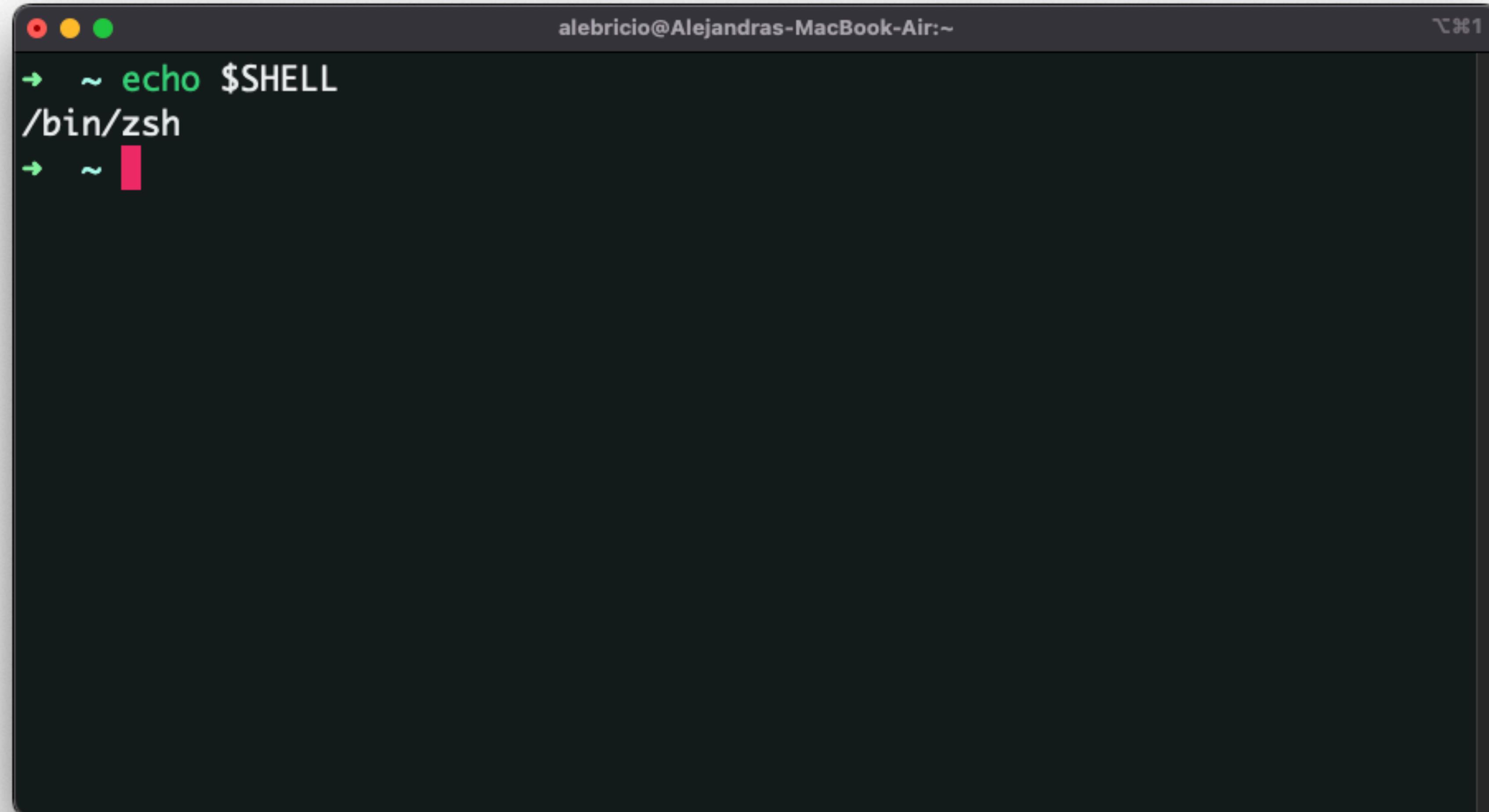
- Site Reliability Engineer @ PayPal
- AWS Solutions Architect – Associate
- AWS Jalisco User Group – Leader
- AWS Community Day Mexico – Organizadora
- Tutora Bootcamp de Ciencias Computacionales @ Código Facilito
 - Redes Computacionales
 - Sistemas Distribuidos

Introducción a Bash Scripting – Temario

- Introducción
- Bash
 - Shell, Terminales
 - Usando bash
 - Navegando bash
 - Otras utilidades (comandos)
- Bash Scripting
 - Sintaxis básica de bash
 - Variables, operadores y estructuras de control

Antes de empezar...

- Para este curso, necesitas estar usando un shell Unix como Bash o ZSH. Si estás en Linux o macOS, no tienes que hacer nada especial. Si estás en Windows, necesitas asegurarte de que no estás ejecutando cmd.exe o PowerShell; puedes usar **Windows Subsystem para Linux** o una máquina virtual Linux para usar herramientas de línea de comandos estilo Unix.
- Para asegurarte de que estás ejecutando un shell apropiado, puedes probar el comando `echo $SHELL`. Si dice algo como `/bin/bash` o `/usr/bin/zsh`, significa que estás ejecutando el programa adecuado.



A screenshot of a macOS terminal window. The title bar shows the user's name and computer: alebricio@Alejandras-MacBook-Air:~. The window contains the following text:

```
→ ~ echo $SHELL  
/bin/zsh  
→ ~
```

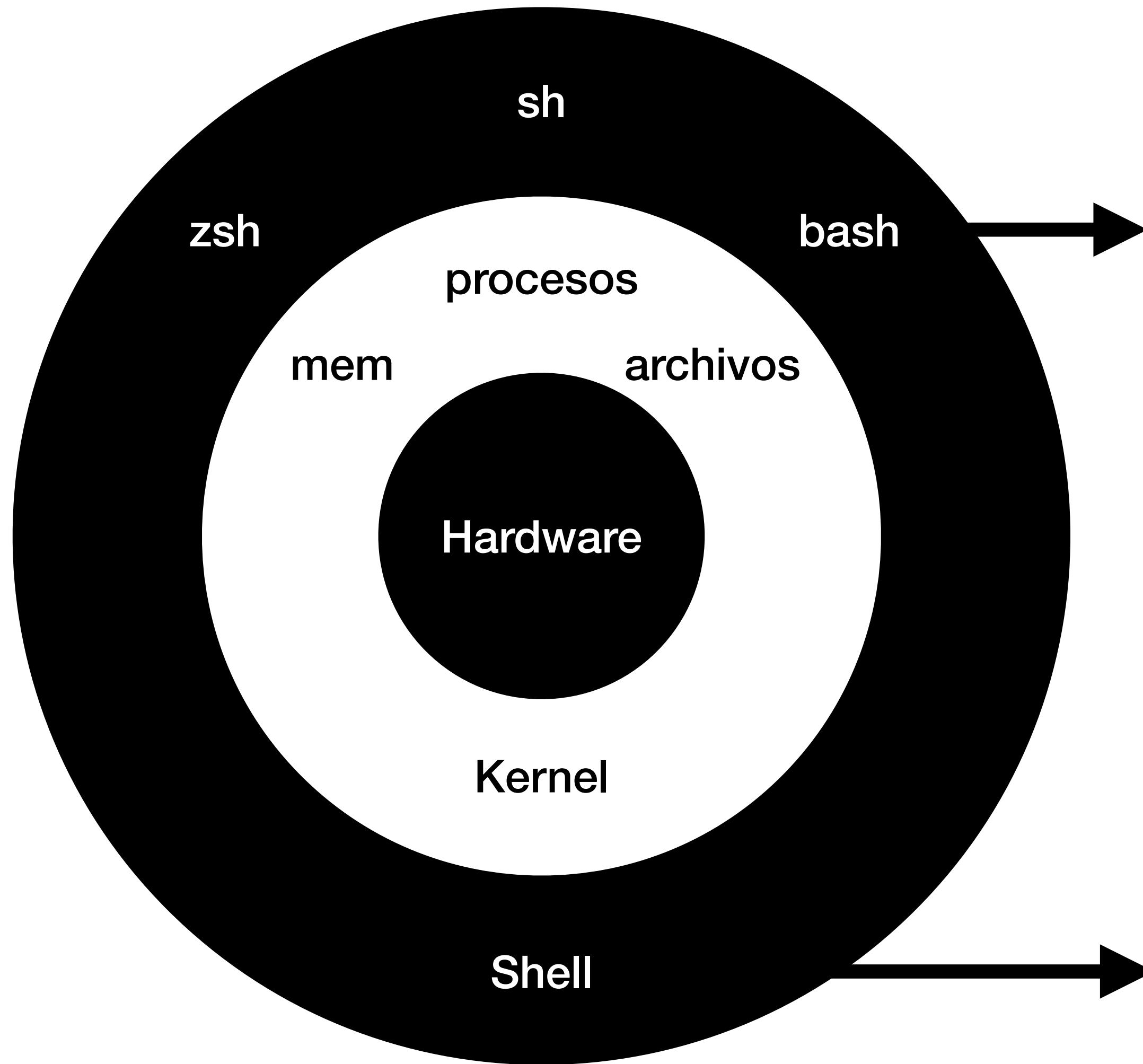
```
alebricio@Alejandras-MacBook-Air:~ % ~ echo "Hola mundo 🌎"  
Hola mundo 🌎  
~ ~
```

¿Qué es un script?

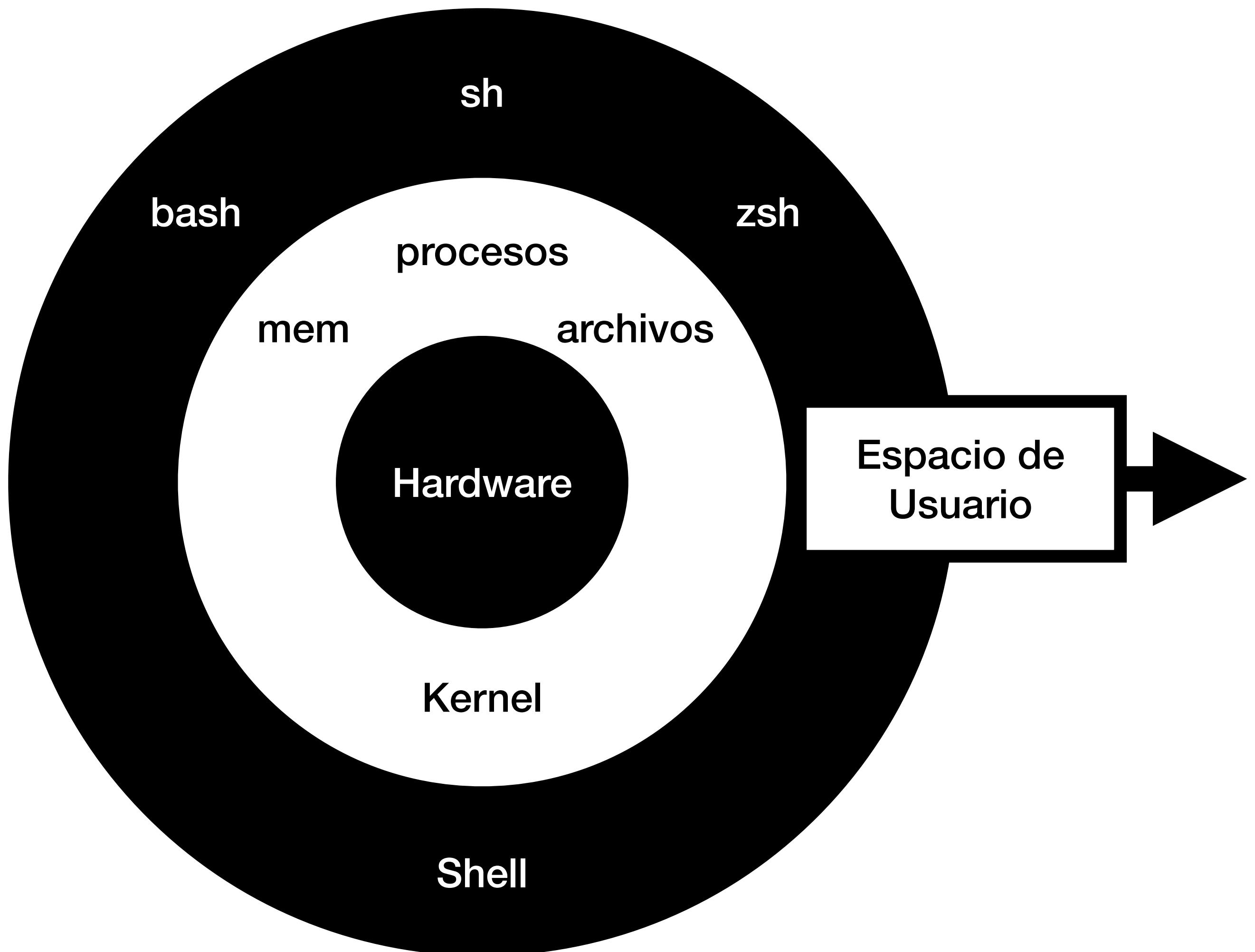
Los scripts ayudan a escribir una secuencia de comandos en un archivo y luego ejecutarlos.

¿Qué es bash?

Linux Shell



Linux Shell



Espacio de Usuario
(User Space)

Aplicación

Aplicación

Aplicación

Bash

Llamadas a sistema

Kernel

- Administración de:
 - Procesos
 - Memoria
- Subsistema de red
- Archivos virtuales

Demo

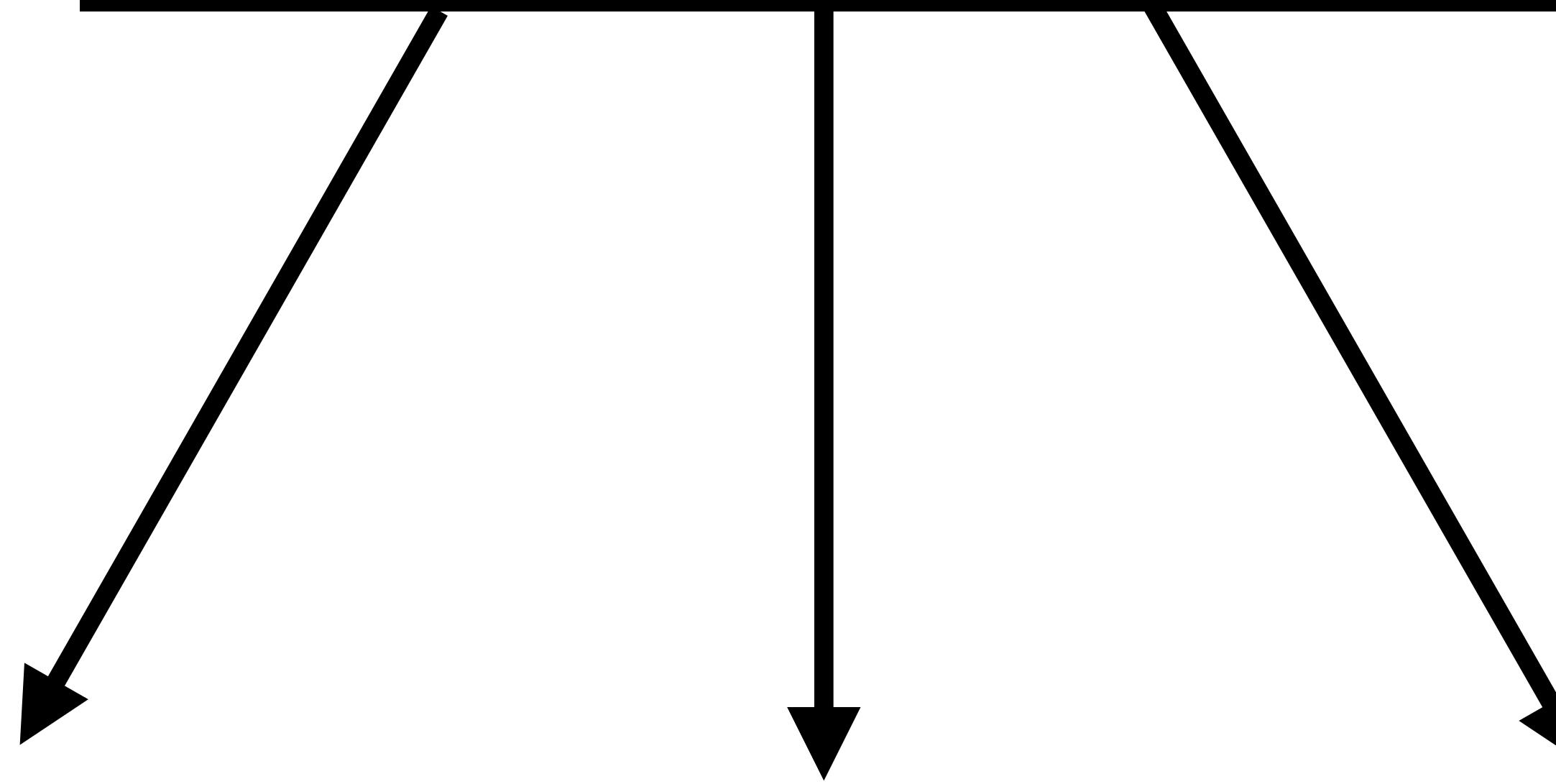
(bash, dtruss, cat, python)

Shell

bash

zsh

sh



Bash

- Mas usuarios y soporte
- Mas compatibilidad con sistemas Linux

Zsh

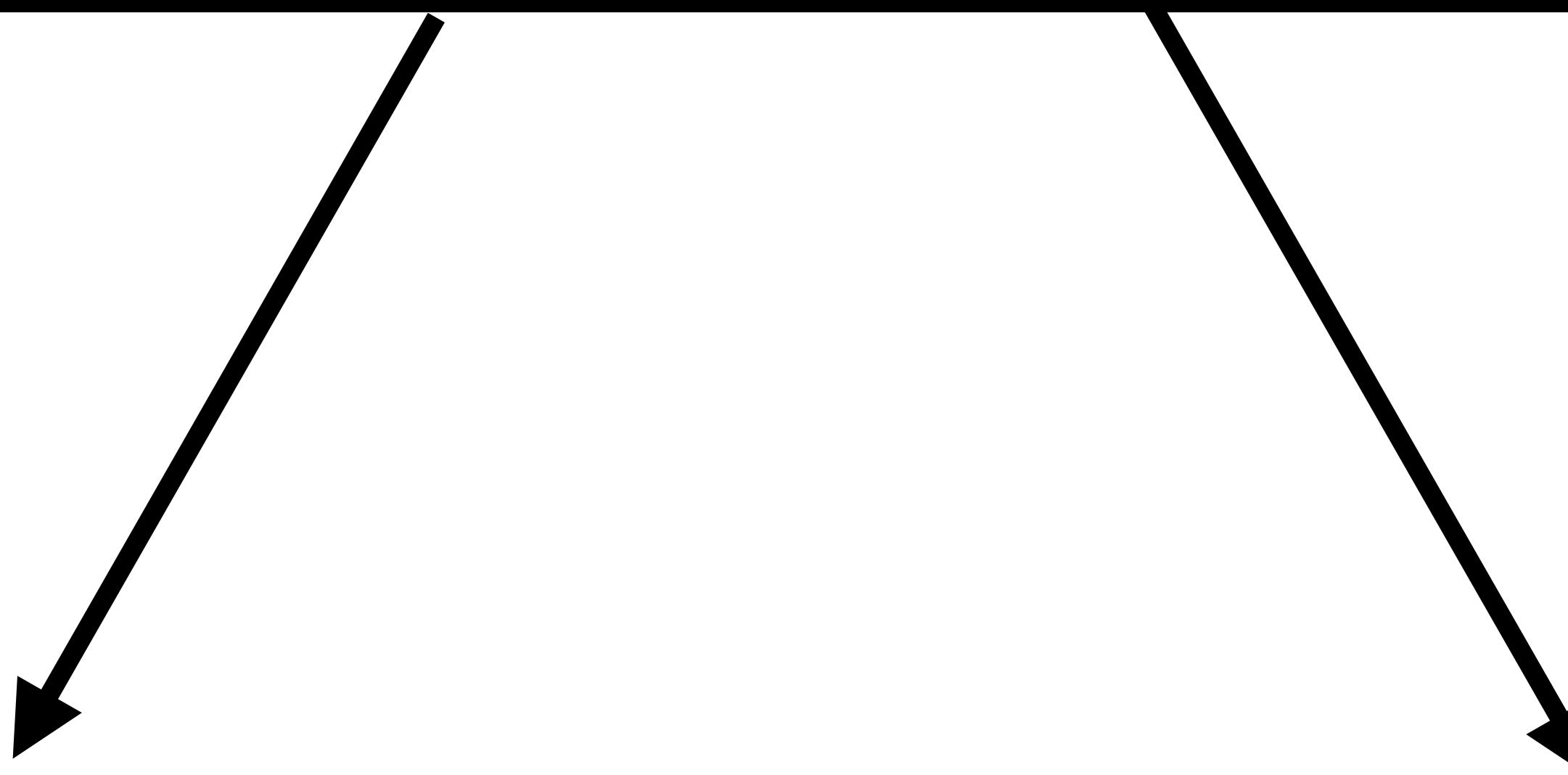
- Mas configurable
 - OhMyZsh
- Tab para navegar entre opciones
- Autocompletado es más rapido
- Syntax highlighting

¿Qué es la terminal?

Terminales

Terminal

iTerm2

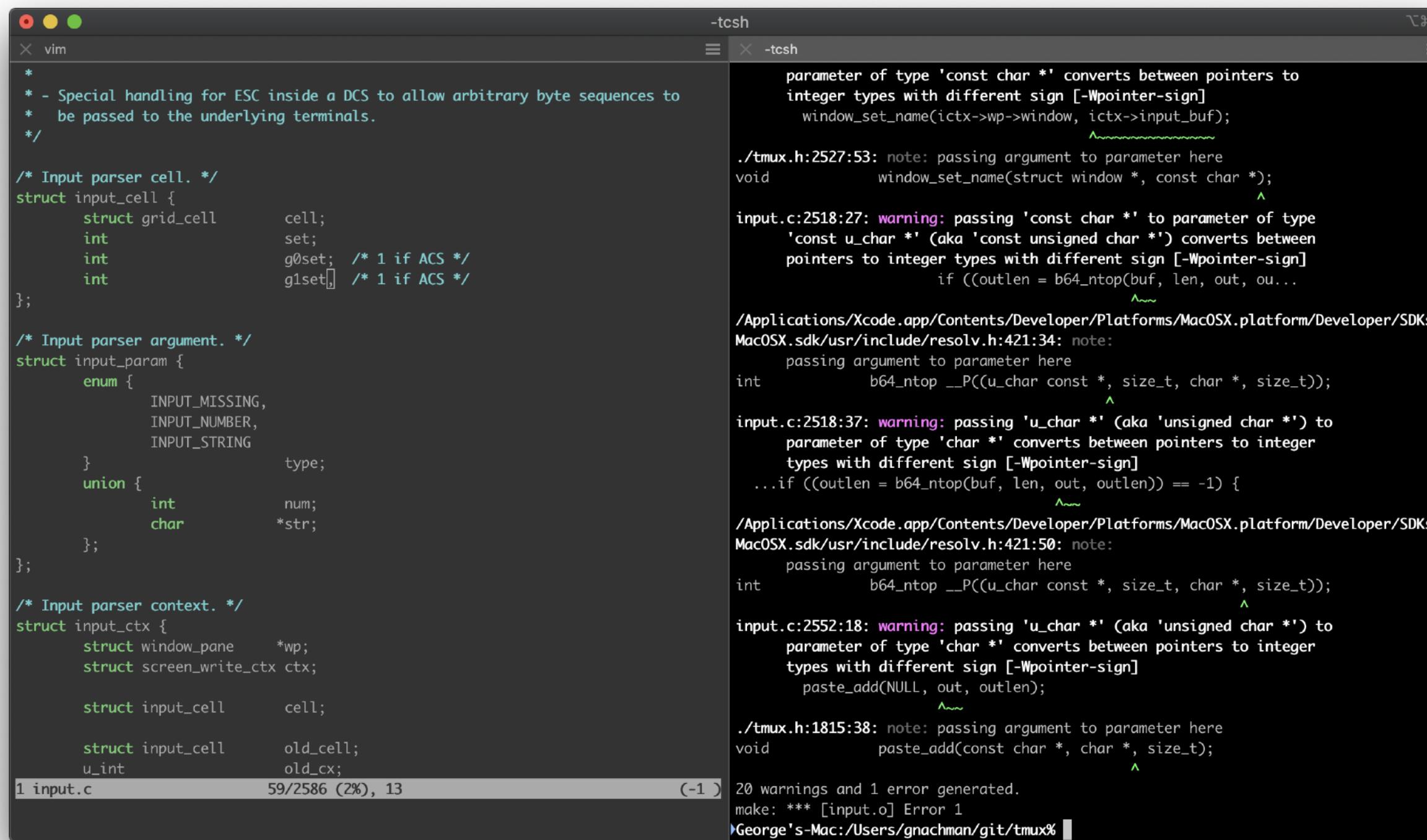




<https://iterm2.com/features.html>

iTerm2 - Paneles

Divida una pestaña en varios paneles, cada uno de los cuales mostrará una sesión diferente. Puede cortar vertical y horizontalmente y crear cualquier número de paneles en cualquier disposición imaginable.



The screenshot shows two iTerm2 panes. The left pane is running vim and displays C code for tmux input parsing. The right pane is running a terminal session (-tcsh) and shows several warning messages from a compiler (likely Clang) about type conversions between pointers and integer types. The warnings are related to the tmux.h header file and involve conversions between const char*, u_char*, and char* types.

```
* - Special handling for ESC inside a DCS to allow arbitrary byte sequences to
* be passed to the underlying terminals.

/* Input parser cell. */
struct input_cell {
    struct grid_cell    cell;
    int                 set;
    int                 g0set; /* 1 if ACS */
    int                 g1set[] /* 1 if ACS */
};

/* Input parser argument. */
struct input_param {
    enum {
        INPUT_MISSING,
        INPUT_NUMBER,
        INPUT_STRING
    } type;
    union {
        int      num;
        char    *str;
    };
};

/* Input parser context. */
struct input_ctx {
    struct window_pane   *wp;
    struct screen_write_ctx ctx;

    struct input_cell    cell;
    struct input_cell    old_cell;
    u_int                old_cx;
};

1 input.c      59/2586 (2%), 13 (-1) 20 warnings and 1 error generated.
make: *** [input.o] Error 1
George's-Mac:/Users/gnachman/git/tmux%
```

iTerm2 - Perfiles

¿Necesita almacenar configuraciones separadas para muchos hosts diferentes? iTerm2 proporciona una base de datos de perfiles etiquetable y con capacidad de búsqueda para que pueda encontrar fácilmente el perfil que está buscando.

The screenshot shows the iTerm2 Profiles window. At the top is a search bar labeled "Search". Below it is a table with columns: Tag Name, Profile Name, Shortcut, and Command. The table contains the following data:

Tag Name	Profile Name	Shortcut	Command
client:bar.com	★ localhost		Login shell
client:foo.com	personal		
devel	Home		Login shell
personal	personal		
production	www01		ssh www01
	client:foo.com, production		
	dev01		sssh dev01
	client:foo.com, devel		
	prod-1		ssh prod-1
	client:bar.com, production		
	host.bar.com		ssh host.bar.com
	client:bar.com, devel		
	Hotkey Window		Login shell

At the bottom are buttons for "< Tags", "Close this window after opening a profile.", "Edit Profiles...", "New Tabs in New Window", "Split Horizontally", "Split Vertically", "New Tab" (highlighted in blue), and "New Window".

The screenshot shows the iTerm2 Profiles window with a search bar containing "client:foo". The results table is identical to the one in the first window, showing profiles like "localhost", "www01", and "dev01" under the "client:foo" tag. The "client:foo" tag is highlighted in yellow in the sidebar.

Tutorial configuración iTerm2

The screenshot shows an iTerm2 window with a single terminal tab titled "1. kevin@100125030: ~/test (zsh)". The terminal window displays a sequence of commands and their outputs:

```
Last login: Mon Jul  1 11:25:50 on ttys001
🏠 ~ ➤ nocmd
zsh: command not found: nocmd
🏠 ~ ➤ mkdir test
🏠 ~ ➤ cd test
📂 ~/test ➤ git init
Initialized empty Git repository in /Users/kevin/test/.git/
📂 ~/test ➤ touch .dirty
📂 ~/test ➤ git add .
📂 ~/test ➤ git checkout -b feature/doink
Switched to a new branch 'feature/doink'
📂 ~/test ➤ git checkout -b feature/doink
```

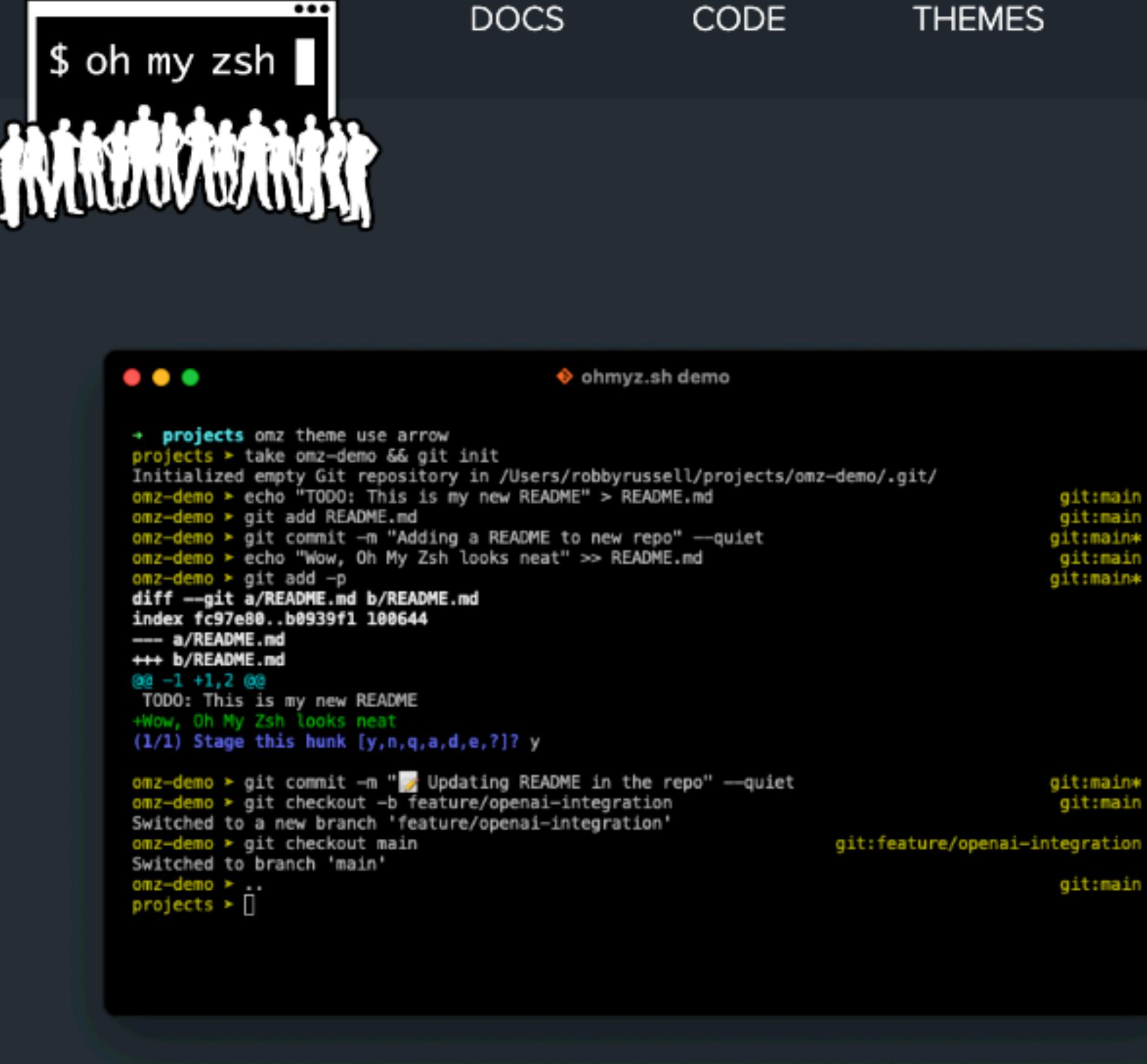
To the right of the terminal window is a sidebar containing several tabs, each with a timestamp and a small icon. The tabs are:

- 9195 11:26:27
- 127 9200 11:26:51
- 9201 11:26:54
- 9202 11:26:56
- 9203 11:26:59
- 9204 11:27:03
- 9207 11:28:54
- 9208 11:29:13

https://www.youtube.com/watch?v=D2PSsnqgBiw&t=990s&ab_channel=QuentinWattTutorials

<https://gist.github.com/kevin-smets/8568070>

oh my zsh



The terminal window shows a user initializing a new Git repository, adding files, committing changes, and switching branches. The output includes color-coded syntax highlighting for commands like `git add` and `git commit`.

DOCS CODE THEMES PLUGINS SHOP SPONSOR

Unleash your terminal like never before.

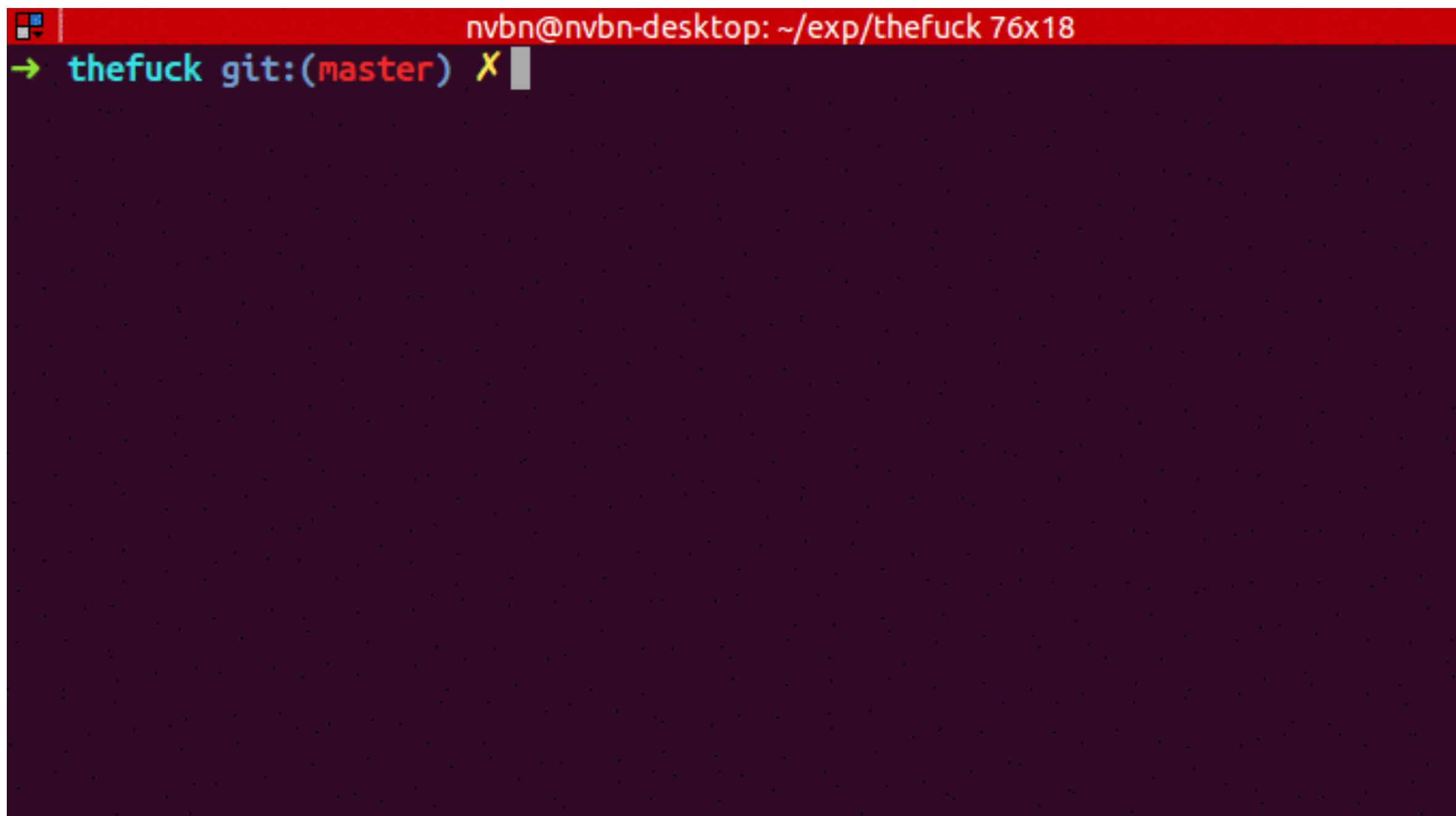
Oh My Zsh is a delightful, open source, community-driven framework for managing your Zsh configuration. It comes bundled with thousands of helpful functions, helpers, plugins, themes, and a few things that make you shout...

"Oh My ZSH!"

[Install oh-my-zsh](#)

 batteries included.

oh my zsh - Plugins

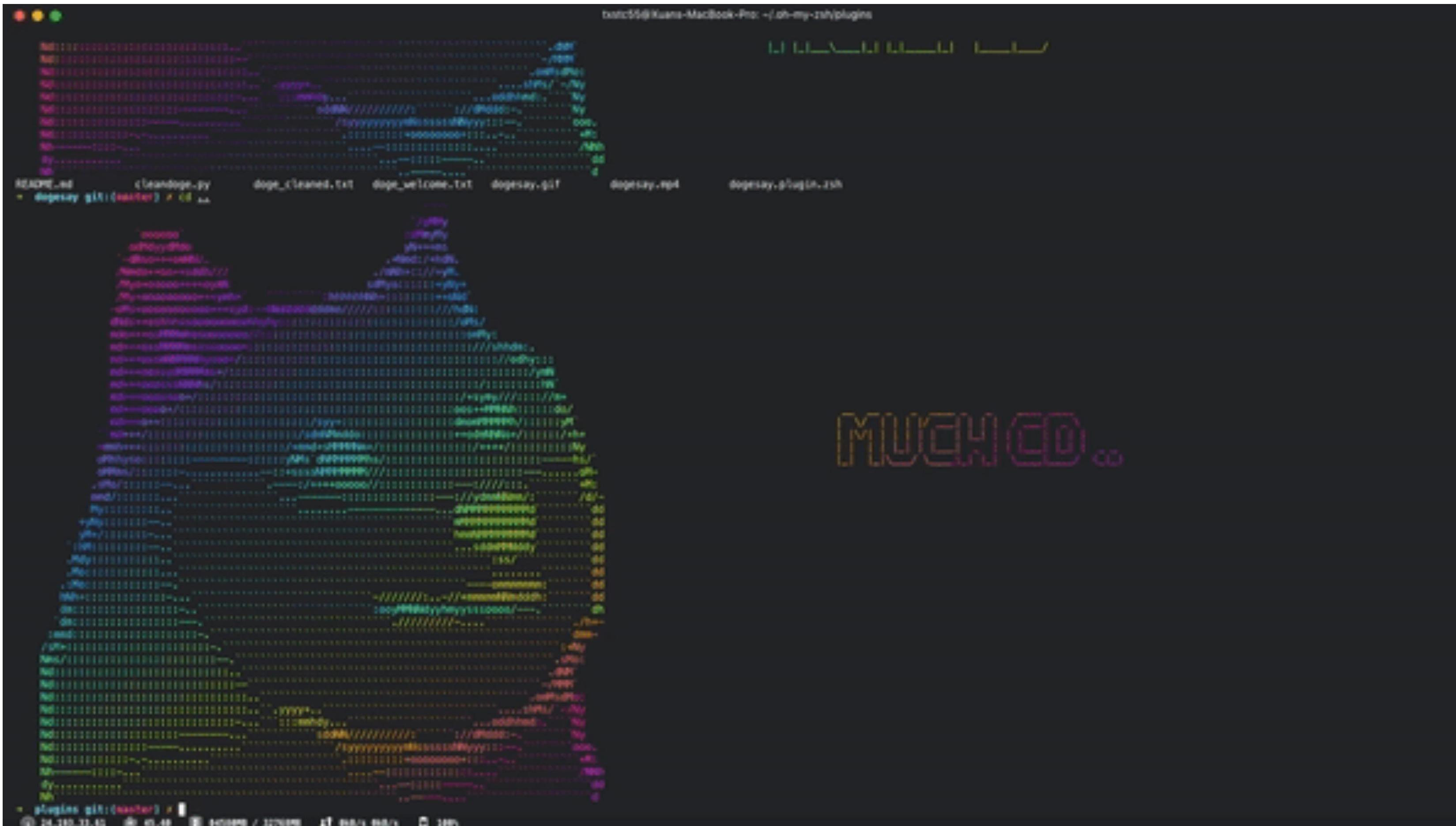


A screenshot of a terminal window titled "nvbn@nvbn-desktop: ~/exp/thefuck 76x18". The prompt shows the user is in a git repository named "thefuck" at the master branch. The terminal displays several examples of the thefuck plugin's behavior:

- The first example shows the plugin fixing a command that included a trailing space: "Before: % echo \$'Hello, world\x21'" and "After: % echo \$'Hello, world\x21'".
- The second example shows it fixing a command with a missing closing quote: "Before: % echo 'Pay \$5 to Joe'" and "After: % echo "Pay \$5 to Joe"".
- The third example shows it fixing a command with a missing closing brace: "Before: % (foo=42)" and "After: % (foo=42)".
- The fourth example shows it fixing a command with a missing closing parenthesis: "Before: % echo 3> /proc/.../vm/drop_caches" and "After: % echo 3> /proc/.../vm/drop_caches".

```
Before: % echo $'Hello, world\x21'  
After: % echo $'Hello, world\x21'  
  
% echo 'Pay $5 to Joe'  
Before: % echo "Pay $5 to Joe"  
After: % echo "Pay $5 to Joe"  
  
Before: % ( foo=42 )  
After: % ( foo=42 )  
  
Before: % echo 3> /proc/.../vm/drop_caches  
After: % echo 3> /proc/.../vm/drop_caches
```

oh my zsh - Plugins



<https://github.com/ohmyzsh/ohmyzsh/wiki/Plugins-Overview>

Inicialización del shell



Archivos de inicialización

Bash

- `.bash_profile`
- `./.bash_profile`
- `...`

Zsh

- `.zshenv`
- `.zprofile`
- `.zshrc`
- `...`

Variables de entorno

- **printenv**

¿Qué es el PATH?

- La variable de entorno global más importante que debes configurar.
- Esta es la variable que le dice al shell bash dónde encontrar diferentes archivos ejecutables y scripts.
- El shell comprobará los directorios listados en la variable PATH para el script que estás intentando encontrar.

Comandos de Bash



Cheatsheet (Hoja de trucos) de comandos

<https://www.geeksforgeeks.org/linux-commands-cheat-sheet/>

Gestión de archivos y directorios

- o **ls**: lista los archivos y directorios.
- o **cd**: cambia el directorio actual.
- o **cp**: copia archivos o directorios.
- o **mv**: mueve o renombra archivos o directorios.
- o **rm**: elimina archivos o directorios.
- o **mkdir**: crea nuevos directorios.

Manipulación de texto

- o **grep**: busca patrones en el texto.
- o **sed**: editor de texto que realiza procesamiento basado en patrones.
- o **awk**: lenguaje de programación diseñado para procesar y analizar archivos de texto.
- o **cat**: concatena y muestra el contenido de archivos.
- o **sort**: ordena las líneas de texto.
- o **uniq**: reporta o filtra líneas repetidas en un archivo.

Administración de procesos

- **ps**: muestra los procesos en ejecución.
- **top**: muestra una lista en tiempo real de los procesos en ejecución.
- **kill**: envía una señal a un proceso para terminarlo.

Gestión del sistema

- o **df**: muestra el uso del disco.
- o **ifconfig/ip**: configura o muestra la configuración de red.
- o **chmod**: cambia los permisos de acceso de archivos o directorios.
- o **chown**: cambia el propietario de archivos o directorios.

Red y comunicaciones

- o **ping**: comprueba la conectividad con otro host.
- o **wget**: descarga archivos desde Internet.
- o **ssh**: inicia sesión en otro computador vía red.
- o **scp**: copia archivos entre hosts en una red.
- o **netstat**: muestra las conexiones de red, tablas de enrutamiento, estadísticas de interfaces, etc.

Administración de paquetes

- **apt-get** (en sistemas basados en Debian): herramienta para manejar paquetes.
- **yum** (en sistemas basados en RedHat): gestor de paquetes.
- **dpkg**: herramienta de gestión de paquetes en Debian.
- **rpm**: gestor de paquetes para RedHat, CentOS, y similares.

Cheatsheet (Hoja de trucos) de comandos

<https://www.geeksforgeeks.org/linux-commands-cheat-sheet/>

Pequeño break ~



Mi primer script de Bash



Lenguaje Bash

- Interpretado
- Extension de archivo: .sh
- Script inicia con: #! /bin/bash
- Permisos de ejecución
- Color de archivo

Mi primer script de Bash

```
touch <miarchivo>
chmod +x <miarchivo>
vi <miarchivo>
```

Definiendo variables



```
#!/bin/bash
# Ejemplo de variable
saludo=Hola
nombre=Cody
echo $saludo $nombre
```

Variables especiales

- o **\$0** representa el nombre del script
- o **\$1 - \$9** los primeros nueve argumentos que se pasan a un script en Bash
- o **\$@** todos los argumentos que se han pasado al script
- o **\$?** la salida del último proceso que se ha ejecutado

Leyendo input de usuarios

```
● ● ●

#!/bin/bash

echo "Ingresa un numero"
read a

echo "Ingresa un numero"
read b

var=$((a+b))
echo $var
```

Operadores Lógicos y Sentencias condicionales

● ● ●

```
if [ condiciones ]
then
    ejecutar comandos
fi
```

● ● ●

```
if [[ condicion ]]
then
    sentencia
elif [[ condicion ]]; then
    sentencia
else
    do hacer esto por defecto
fi
```

● ● ●

```
#!/bin/bash

read x
read y

if [ $x -gt $y ]
then
    echo X es mayor que Y
elif [ $x -lt $y ]
then
    echo X es menor que Y
elif [ $x -eq $y ]
then
    echo X es igual que Y
fi
```

For Loops



```
#!/bin/bash

for i in {1..5}
do
    echo $i
done
```



```
#!/bin/bash

for X in rojo verde azul
do
    echo $X
done
```

While Loops

```
#!/bin/bash

LINEA=1

while read -r LINEA
do
    echo "$LINEA: $LINEA_ACTUAL"
    ((LINE++))
done < "archivo_ejemplo.txt"
```

¿Preguntas?



@alebricio

Tarea



@alebricio

¡ Gracias !



@alebricio