

>_ WHOAMI?


0sGa or os24





- CTF Player | Developer
- A student from NYUST
- btw I use Vim

more on <https://osga.dev>



>_ 今天要做啥？


- 已經熟悉 Linux 操作了 

- 但有一些小問題 
 -  為什麼 `git status` 要打那麼長？不能改短一點嗎？
 -  為什麼連上遠端主機要打一大串 IP + 帳號？
 -  還有什麼工具有助於日常使用呢？

 DEMO 一下我ㄉ日常好ㄌ

 添加一些設定，省去平常糟做得麻煩 

 如果講太快的話我可能會帶到下禮拜的內容

 可能今天東西比較雜，介紹一下小設定 / 工具可以便於你一些日常操作，下周會著重在使用上。

>_ Slido

有任何問題都歡迎提問！

CODE: 9025087



>_ 前情提要

預先善其事 必先利其器

- 今天操作會用到很多需要編輯設定的部分
- 好的文字編輯器是非常重要的 ✅

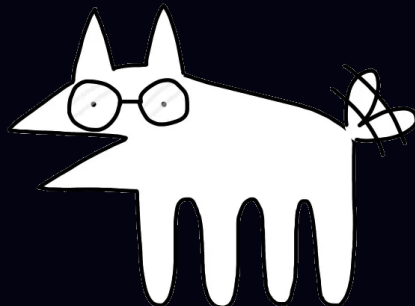


選擇你的派系 🍷

反正我們今天要用 VIM

>_ 為什麼是 Vim 不是 nano

- 不需滑鼠，快速移動、複製、選取文字
- 可以搭配 NeoVim 等擴展更多功能
- 生態系完整，較多人使用
- 還有很多你用了就知道



用 nano 的都是男娘



>_ Welcome join Vim world

- 今天只會先教大家基本操作
- 其餘設定下禮拜會在教各位一些更詳細的操作

>_ Welcome join Vim world

■ how to install VIM?

- VIM 基本上已經是 Linux 標配的套件

■ How to use VIM

- 開啟 VIM

```
vim
```

```
vi
```

- 編輯檔案

```
vim <file>
```

```
vi <file>
```

▤ 如果檔案不存在，編輯即創建！

▤ 為什麼下面一個 vi 也可以使用到 Vim?

>_ Welcome join Vim world

🇺🇸 為什麼下面一個 vi 也可以使用到 Vim?








- Vi 是最早期的 Unix 編輯器之一
- 出現在 1970s，是 POSIX 標準要求的基本工具。
- Vim (Vi IMproved) 於 1991 年推出
- 為 Vi 的加強版，支援更多功能（如多層 undo、syntax highlight）。
- Vim 保留向下相容性
- 使用方式與 Vi 幾乎相同，因此可安全替代。
- Linux 發行版為了統一維護與使用體驗
- 多數預設將 vi 設為 Vim 的 symlink (例如 /usr/bin/vi -> /usr/bin/vim) 。
- 便於使用者獲得更多功能而不需額外安裝
- 提供 Vi 介面熟悉度的同時，享有 Vim 強大功能。









>_ Welcome join Vim world

Vim 操作入門指南

模式介紹 (Modes)

模式	進入方式	用途說明
Normal		預設開啟即為此模式 移動游標、刪除、複製等操作
Insert	 、  、  等	編輯文字 (像一般編輯器一樣)
Visual	 、  、 	選取文字區塊進行複製、刪除等操作
Command		執行指令 (如儲存、離開)

模式切換

操作	效果
	插入游標前
	插入游標後
	在下方開新行並插入
	回到 Normal 模式
	Visual 模式 (選字元)
	Command 模式

>_ Welcome join Vim world


Vim 操作入門指南

常用移動指令 (Normal 模式)

指令	功能
h	向左移動一格
l	向右移動一格
j	向下移動一行
k	向上移動一行
w	移動到下一個單字開頭
0	移到行首
\$	移到行尾
gg	移到檔案最上方
G	移到檔案最下方

編輯與操作 (Normal 模式)

指令	功能
x	刪除游標所在字元
dd	刪除整行
yy	複製整行
p	貼上 (在後)
u	復原
Ctrl + r	反復原
	重複上一個編輯動作

 <https://vim.rtorr.com/>

>_ Welcome join Vim world

📖 Vim 操作入門指南

📄 儲存與離開 (Command 模式)

//怎麼退出RRR

進入 Command 模式先按 `|`，再輸入以下指令：

指令	功能
<code>:w</code>	儲存
<code>:q</code>	離開
<code>:wq</code>	儲存並離開
<code>:q!</code>	不儲存強制離開



<https://github.com/hakluke/how-to-exit-vim>

🔪 練習

創建一個 `*.py`

使用 Vim 寫一個簡單的 `print("hello world")`

1. 開啟檔案：

```
vim test.py
```

2. 嘗試：

◦ 使用 `|` 編輯文字，再輸入

```
print('hello world')
```

◦ 按 `ESC` 回到 Normal Mode，用 `:wq` 儲存離開

>_ Welcome join Vim world



🔮 小魔法讓 Vim 更好用一點

Verrrrrrrrrrrry Goooooooooooooooooooo 你現在是一個合格的 Vim users 了

趕緊推坑你身邊的朋友入教

但現在的 Vim 看起來還很陽春

📝 添加一些小設定讓他更方便！

```
/etc/vim/vimrc
```

Vim 勿設定檔

🇺🇸 進入編輯

```
sudo vim /etc/vim/vimrc
```

可以都多善用你勿 TAB

📄 FYI

- vimrc 是 Vim 啟動時預設會讀取的設定檔之一（屬於全域層級）。
- 作用於 所有使用者，除非被使用者的個人設定（如 ~/.vimrc）覆蓋。
- 由系統管理員（如 root）配置，提供預設行為。

>_ Welcome join Vim world

🔪 小魔法讓 Vim 更好用一點

假設說今天有一個程式

```
def hello(name):  
    print(f"hello {name}")  
  
name = input(input user name: )  
hello(name)
```

test.py

```
" All Debian-specific settings are defined in $VIMRUNTIME/debian.vim and  
" sourced by the call to :runtime you can find below. If you wish to change  
" any of those settings, you should do it in this file or  
" /etc/vim/vimrc.local, since debian.vim will be overwritten everytime an  
" upgrade of the vim packages is performed. It is recommended to make changes  
" after sourcing debian.vim so your settings take precedence.  
  
runtime! debian.vim  
  
" Uncomment the next line to make Vim more Vi-compatible  
" NOTE: debian.vim sets 'nocompatible'. Setting 'compatible' changes  
" numerous options, so any other options should be set AFTER changing  
" 'compatible'.  
"set compatible
```

/etc/vim/vimrc 的某一段

>_ Welcome join Vim world

🔪 小魔法讓 Vim 更好用一點

```
" All Debian-specific settings are defined in $VIMRUNTIME/debian.vim and
" sourced by the call to :runtime you can find below.  If you wish to change
" any of those settings, you should do it in this file or
" /etc/vim/vimrc.local, since debian.vim will be overwritten everytime an
" upgrade of the vim packages is performed. It is recommended to make changes
" after sourcing debian.vim so your settings take precedence.
```

```
runtime! debian.vim
set ai nu
" Uncomment the next line to make Vim more Vi-compatible
" NOTE: debian.vim sets 'nocompatible'.  Setting 'compatible' changes
" numerous options, so any other options should be set AFTER changing
" 'compatible'.
"set compatible
```

/etc/vim/vimrc 的某一段

加上了 `set ai nu`

```
1 def hello(name):
2     print(f"hello {name}")
3
4 name = input('input user name: ')
5 hello(name)
```

多出酷酷的數字了

可以用 `g` 方便跳到對應行數

>_ Welcome join Vim world

👉小魔法讓 Vim 更好用一點

當然還有其他參數

指令	說明
<code>set number</code>	顯示行號
<code>set relativenumber</code>	顯示相對行號（搭配移動更方便）
<code>syntax on</code>	啟用語法高亮
<code>set tabstop=4</code>	設定 tab 寬度為 4 空格
<code>set shiftwidth=4</code>	設定縮排寬度為 4 空格（影響 <code>>></code> 、 <code><<</code> ）
<code>set softtabstop=4</code>	設定按一次 Tab 鍵產生的空格數量
<code>set expandtab</code>	使用空格取代 Tab 字元
<code>set autoindent</code>	啟用自動縮排
<code>set smartindent</code>	根據語法自動判斷縮排（適合程式碼）
<code>set nowrap</code>	關閉自動換行（長行不換行）
<code>set scrolloff=5</code>	游標靠近上下邊界時預留 5 行可視範圍
<code>set cursorline</code>	突出顯示目前游標所在行
<code>set incsearch</code>	即時搜尋（輸入時立即高亮）
<code>set ignorecase</code>	搜尋時忽略大小寫
<code>set smartcase</code>	若搜尋中含有大寫則開啟大小寫區分
<code>set hlsearch</code>	高亮顯示所有搜尋結果
<code>set clipboard=unnamedplus</code>	使用系統剪貼簿（可與外部系統互通）
<code>set mouse=a</code>	啟用滑鼠操作（選取、拖曳等）
<code>set background=dark</code>	適合深色背景主題（自動調整語法配色）
<code>filetype plugin indent on</code>	啟用語言檔案偵測、自動縮排與對應插件支援

下禮拜會再帶大家多玩玩！

先知道怎麼編輯寫入退出就好

>_ 將一大坨的指令設定快捷

- 「我們每天都在打一樣的指令：git status, ls -la, cd ../../..，你會不會覺得煩？」
- 你會常用哪些重複的指令？

👉 我們可以在 ~/.bashrc or ~/.zshrc 下設定將指令設定快捷



```
ping 8.8.8.8
```

可以使用 **alias** 將整串指令自定義成自己的習慣

```
alias p8='ping 8.8.8.8'
```


👉 要寫在哪呢？

>_ 將一大坨的指令設定快捷

每個 shell 在家目錄底下都有一個 `*rc`



```
.zshrc .bashrc .fishrc
```

 查看現在使用哪個 SHELL

```
echo $SHELL
```

```
(osga@MacKali) [~]  
$ echo $SHELL  
/usr/bin/zsh
```

✔ zsh

>_ 將一大坨的指令設定快捷

進入設定檔

```
vi ~/.zshrc
```

在空白處加上



```
alias p8='ping 8.8.8.8'
```

套用 / 生效設定

```
source ~/.zshrc
```

使用

```
(osga@MacKali) [~]  
$ p8  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=12.2 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=128 time=8.24 ms
```

  動手試試時間

>_ 將一大坨的指令設定快捷

如果今天是複雜的指令怎麼辦 🤔

- 需要輸入到參數
- 自己寫一個 command ✅

ex.

linux

- 一個我之前方便連線到不同機器的工具
- ssh 到固定主機

```
1 #!/bin/bash
2
3 if [ "${1}" = "rk" ]; then
4     ssh osga@10.8.0.5
5 elif [ "${1}" = "lk" ]; then
6     ssh osga@192.168.200.131
7 elif [ "${1}" = "x86" ]; then
8     ssh osga@192.168.64.15
9 else
10     echo "ERROR"
11 fi
```

```
09:41:27 下午 ☺
linux lk
Linux MacKali 6.12.13-arm64 #1 SMP Kali 6.12.13-1kali1 (2025-02-11) aarch64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 16 21:20:10 2025 from 192.168.200.1
(osga@MacKali) [~]
$
```

>_ 將一大坨的指令設定快捷

🇺🇸 如果今天是複雜的指令怎麼辦 🤔

將檔案製作完成後，可以放到

```
/usr/local/bin
```

🇺🇸 ✍️ 動手試試時間

🇺🇸 1.

寫一個指令叫做 `hello`

執行指令會輸出 `hello world`

🇺🇸 2.

承上題.

`hello` 後面可以接字串

執行指令會輸出 `hello ${string}`

>_ 將一大坨的指令設定快捷

如果今天是複雜的指令怎麼辦 🤔

動手試試時間

1.

寫一個指令叫做 `hello`

執行指令會輸出 `hello world`

- 創建一個 shell code 叫做 `hello`

```
vi hello
```

- 寫 shell code

```
#!/bin/bash
```

```
echo hello world
```

- 儲存退出，並且將檔案移至 `/usr/local/bin/`

```
sudo mv hello /usr/local/bin
```

可能有權限問題 建議加上 `sudo`

>_ 將一大坨的指令設定快捷

🇺🇸 如果今天是複雜的指令怎麼辦 🤔

📝 ✍️ 動手試試時間

🇺🇸 2.

承上題.

hello 後面可以接字串

執行指令會輸出 `hello ${string}`

- 創建一個 shell code 叫做 `hello`

```
vi hello
```

- 寫 shell code

```
#!/bin/bash
```

```
echo hello ${1}
```

- 儲存退出，並且將檔案移至 `/usr/local/bin/`

```
sudo mv hello /usr/local/bin
```

可能有權限問題 建議加上 `sudo`


🇺🇸 用 shell code 寫ㄉ一個 sideproject <https://github.com/osga24/Quick-Jump>

>_ ssh 免密碼連線

當你使用進一台機器時...

```
1 └─(osga@MacKali)~]
2 └─$ ssh osga@localhost
3 osga@localhost's password:
4 Linux MacKali 6.12.13-arm64 #1 SMP Kali 6.12.13-1kali1 (2025-02-11) aarch64
5
6 The programs included with the Kali GNU/Linux system are free software;
7 the exact distribution terms for each program are described in the
8 individual files in /usr/share/doc/*/copyright.
9
10 Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
11 permitted by applicable law.
12 Last login: Fri May 16 22:03:34 2025 from 192.168.200.1
13
14 └─(osga@MacKali)~]
15 └─$
```

如果這台機器你很常連的話呢...

 @@ 每次都要重複輸入密碼

 使用金鑰自動驗證 

>_ ssh 免密碼連線

1. 創建金鑰

```
(osga@MacKali) [~]  
$ ssh-keygen
```

```
Generating public/private ed25519 key pair.  
Enter file in which to save the key (/home/osga/.ssh/id_ed25519):  
// 產生一對 SSH 金鑰（公鑰 + 私鑰），使用 ed25519 演算法  
Enter passphrase for "/home/osga/.ssh/id_ed25519" (empty for no passphrase):  
// 選擇儲存檔案（預設 ~/.ssh）  
Enter same passphrase again:  
// 設定 金鑰的密碼保護，如果你輸入密碼，這把私鑰就只有在提供該密碼後才能使用  
Your identification has been saved in /home/osga/.ssh/id_ed25519  
Your public key has been saved in /home/osga/.ssh/id_ed25519.pub  
// 成功儲存私鑰公鑰  
The key fingerprint is:  
SHA256:veGoRu0000qpWChoftXpXFgQC9qlGPA03CllnaF4oJk osga@MacKali  
// 這是金鑰的指紋（fingerprint），用來在安全交換時確認是否被竄改過  
The key's randomart image is:  
+--[ED25519 256]--+  
| .o=o+o++ |  
| *oXo=+ |  
| E =. = .. |  
| . o |  
| ..S o |  
|.o ..=++ o |  
|+.o ..=oooo |  
|+ o o.++ |  
| .. ..o. . |  
+----[SHA256]-----+  
// 金鑰的 視覺指紋（visual hash），目的是讓你在一堆金鑰中快速「肉眼比對」
```

■ 這邊先全部按 enter 作為範例

>_ ssh 免密碼連線

```
└─(osga@MacKali)-[~]
```

```
└─$ cd .ssh
```

```
id_ed25519 id_ed25519.pub known_hosts known_hosts.old
```

可以將你的 *.pub 丟給別人

我們先在本機 Try Try

>_ ssh 免密碼連線

```
sudo vi /etc/ssh/sshd_config
```

```
35. #MaxAuthTries 6
36. #MaxSessions 10
37.
38. #PubkeyAuthentication yes
39.
40. # Expect .ssh/authorized_keys2 to be disregarded by default in future.
41. AuthorizedKeysFile      .ssh/authorized_keys .ssh/osga24.keys
42. #AuthorizedPrincipalsFile none
43.
44. #AuthorizedKeysCommand none
45. #AuthorizedKeysCommandUser nobody
```


>_ ssh 免密碼連線

```
sudo vi /etc/ssh/sshd_config
```

```
35. #MaxAuthTries 6
36. #MaxSessions 10
37.
38. #PubkeyAuthentication yes
39.
40. # Expect .ssh/authorized_keys2 to be disregarded by default in future.
41. AuthorizedKeysFile      .ssh/id_ed25519.pub .ssh/osga24.keys
42. #AuthorizedPrincipalsFile none
43.
44. #AuthorizedKeysCommand none
45. #AuthorizedKeysCommandUser nobody
```

 儲存退出 重啟服務

```
sudo service ssh restart
```

 如果沒有開啟的話就

```
sudo service ssh start
```

>_ ssh 免密碼連線

現在重先連線

```
(osga@MacKali) [~]
$ ssh osga@localhost
Linux MacKali 6.12.13-arm64 #1 SMP Kali 6.12.13-1kali1 (2025-02-11) aarch64


The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 16 23:08:32 2025 from 192.168.200.1
(osga@MacKali) [~]
$
```

 動手試試時間

和你旁邊的夥伴

互相丟公鑰給對方 ssh

 可以使用 scp 互相丟公鑰給對方

>_ 工具分享

一些好玩的工具可以進化日常的使用或優化其他指令效果

- cowsay
 - 可以讓一隻牛說話
- nyan-cat
 - 打了就知道ㄌ
- cmatrix
 - 裝完會變駭客
- Quick-Jump
 - 快速在不同目錄跳躍 安麗一下自己工具
- bat
 - `batter cat`
- htop
 - 查看 CPU、RAM、Process 的互動介面，比 top 更清楚
- neofetch
 - 顯示系統資訊
- thefuck
 - 打錯指令自動修正

-
- 下次課程會著重在
 - tmux
 - neovim
 - yazi

推薦大家自己安裝看看或尋找 / 寫符合自己要求的套件

