

You don't need Plugin

for some case

Long Live Command Line

(full version)

jaeyeol.lee@hey.com

Jaeyeol Lee

- **Moderator of vim.kr**
at Discord, known as **kokoko.kojima**
- Developed **mastodon.nvim**
(mastodon client for neovim)
- I gave a few talks about Neovim in South Korea.
 - “**Neovim으로 생산성 퀀텀점프하기**”
 - <https://slides.kodingwarrior.dev>



We will talk about...

- 제 목적에 맞게 CLI를 활용하자
- CLI를 직접 짜는 방법도 살펴보자
- 플러그인 없이도 어떻게 워크플로우를 개선할 수 있는지 살펴보자

우리는 정말로 플러그인이 필요할까요?

어떤건 워크플로우 개선에 큰 도움이 됩니다.

- oil.nvim : File system navigation, Vim 조작하듯이 파일 복사/생성/삭제
- nvim-spectre : Grep & Replace
- leap.nvim : 커서를 빠르게 이동

어떤 플러그인은 정말 필요합니다

- Language Server ----- 자동 완성, diagnostics, go to definition
- Treesitter ----- semantic highlighting
- telescope.nvim ----- Neovim과 통합된 쿼리 시스템
- nvim-cmp
- snippet

플러그인 작성도 어렵지 않아요

- **플러그인 작성이 여러모로 도움이 됩니다**

- 플러그인을 개발하다보면, Neovim이 어떻게 동작하는지 인사이트를 얻어요.
- 플러그인을 개발하다보면, dotfiles를 효과적으로 구성하는 요령을 얻어요.
- 플러그인을 개발하다보면, neovim에서 제공하는 API를 잘 이해하게 되어요.

- **하지만, 시간을 많이 잡아먹어요**

어쩔 때는

플러그인이 그닥 필요하지 않을 수도 있어요

어떤 플러그인은 그닥 직관적이진 않아요

- 키보드에서 손을 떼고 싶지 않다한들...
터미널에서 벗어나고 싶지 않다 한들...
Vim 그 자체로는 한계가 있을 때도 있어요
- For GUI engine(GTK, Web browser, ...)
 - Many UI/UX support
 - Drag&Drop
 - Click, Click, Click,
 - 인지부하를 줄여주는 여러가지 시각화 라이브러리...
- For TUI...? 🤪🤪🤪

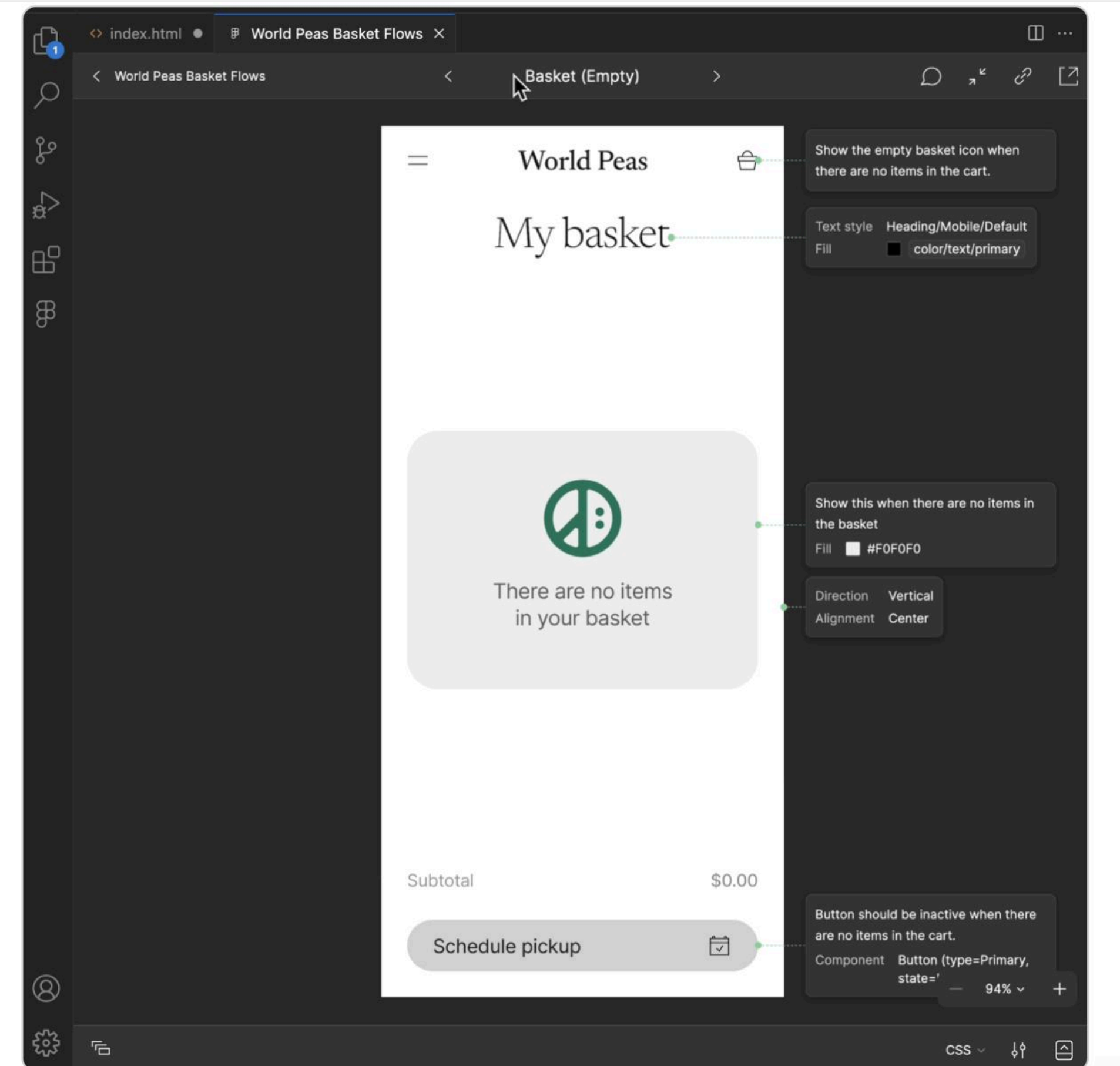
플러그인 만드는 것 자체가 어려울 수도 있어요

- 개발하는 것 자체의 복잡도
 - paradigm : Imperative > procedural
 - mental model : (GUI) tree > (TUI) text-base 2d array
- 개발하는 것 자체의 난이도 (for utilizing Vim as TUI Engine)
- 리소스는요?
 - 튜토리얼이 사실상 없거나..
 - 다른 사람이 만든거 참고하거나...

플러그인 개발 자체가 리소스를 잡아먹어요

- Vim 플러그인 개발 자체는 2차원 문자열 배열을 가지고 노는 것에 가까워요.
문자열 알고리즘에 능하다면 어렵지 않게 짤 수 있어요.
- 플러그인을 개발한다는 것 자체가 고려할 요소가 많습니다.
 - **UX가 굉장히 중요해요** : 직관적이지도 않거나 워크플로우를 최적화하는데 도움이 되지 않는다면, 안 짤 것만도 못할 수 있습니다.
 - **개발하는 것 자체의 난이도도 있어요** : telescope, nvim-cmp랑 통합한다면?
 - **개발 자체가 시간이 걸려요**
 - 특정 목적으로 플러그인 짜는데 쓰는 시간이 가치 있을까요?

VSCode figma integration

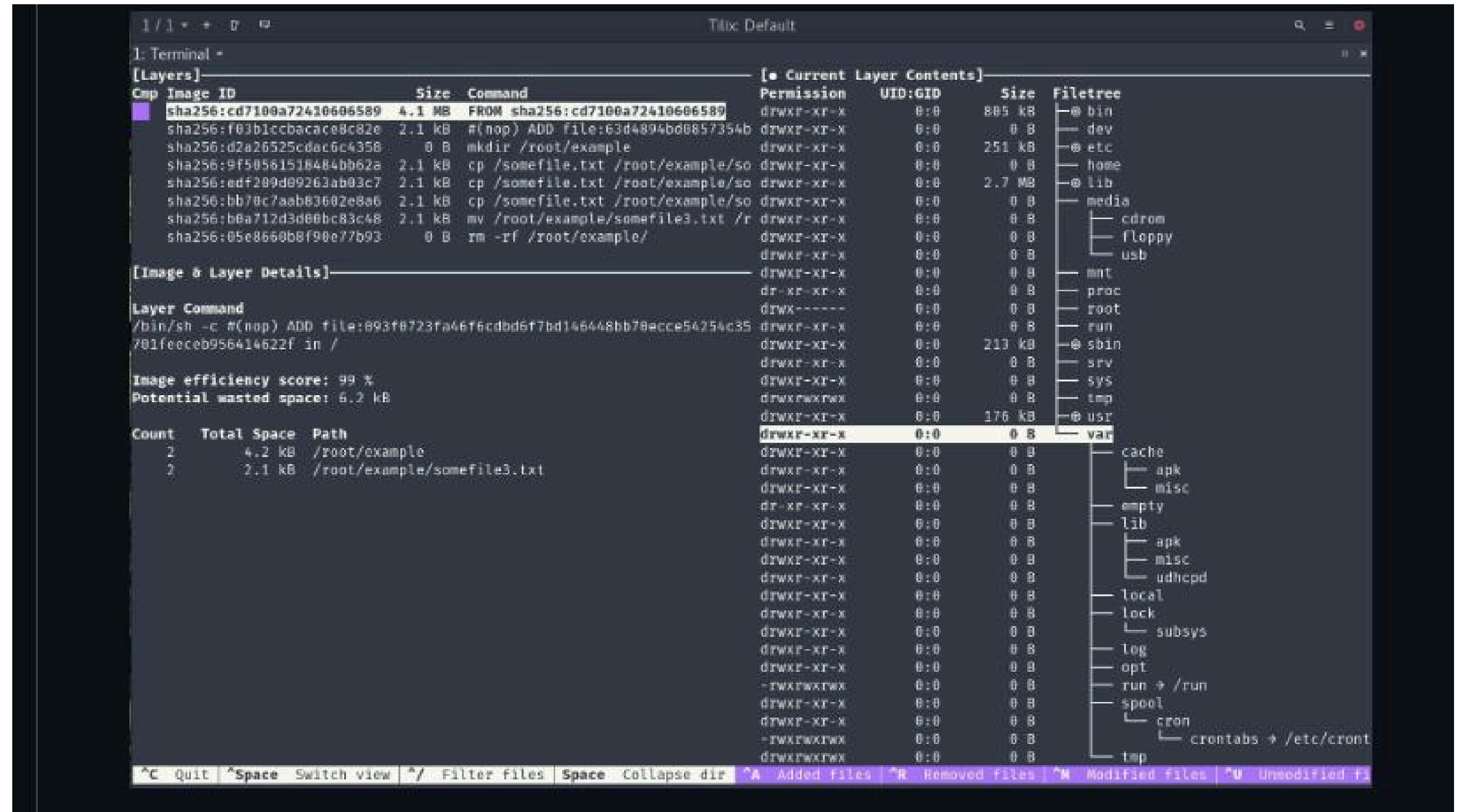


어떤 플러그인은 CNI 도구를 감싼 것에 불과해요

- Vim-fugitive는 사실상 git 커맨드를 vim api로 감싼 것 (통합이 엄청 잘된 케이스)
- octo.nvim는 사실상 gh 커맨드를 nvim api로 감싼 것
- 몇몇 사람들은 굳이 플러그인으로 짤 필요를 느끼지 못 합니다.

이미 좋은 CLI 도구는 충분히 많습니다

- ex) dive - docker image explorer



The screenshot displays the Dive CLI interface for exploring a Docker image. The main window is titled "Tmux Default" and shows a terminal view with the following sections:

- [Layers]**: A table listing image layers with columns for Cmp, Image ID, Size, Command, Permission, UID:GID, and Size. The current layer is highlighted in purple.
- [Image & Layer Details]**: A section showing the layer command and image efficiency score (99%).
- Filetree**: A tree view of the file system structure, including directories like bin, dev, etc, home, lib, media, mnt, proc, root, run, sbin, srv, sys, tmp, and usr.
- Count Table**: A summary table showing the count of files and their total space.
- Footer**: A navigation bar with keyboard shortcuts for various actions like Quit, Space, Switch view, Filter files, Collapse dir, Added files, Removed files, Modified files, and Unmodified files.

Cmp	Image ID	Size	Command	Permission	UID:GID	Size
sha256:cd7100a72410606589	4.1 MB	FROM sha256:cd7100a72410606589	drwxr-xr-x	0:0	805 kB	
sha256:f03b1ccbacace8c82e	2.1 kB	#(nop) ADD file:63d4894bd0857354b	drwxr-xr-x	0:0	0 B	
sha256:d2a26525cdac6c4358	0 B	mkdir /root/example	drwxr-xr-x	0:0	251 kB	
sha256:9f50581518484bb62a	2.1 kB	cp /somefile.txt /root/example/so	drwxr-xr-x	0:0	0 B	
sha256:edf209d09263ab03c7	2.1 kB	cp /somefile.txt /root/example/so	drwxr-xr-x	0:0	2.7 MB	
sha256:bb70c7a0b03602e8a6	2.1 kB	cp /somefile.txt /root/example/so	drwxr-xr-x	0:0	0 B	
sha256:b0a712d3d00bc83c48	2.1 kB	mv /root/example/somefile3.txt /r	drwxr-xr-x	0:0	0 B	
sha256:05e8660b0f90e77b93	0 B	rm -rf /root/example/	drwxr-xr-x	0:0	0 B	

Count	Total Space	Path
2	4.2 kB	/root/example
2	2.1 kB	/root/example/somefile3.txt

근본으로 돌아가기

근본으로 돌아가기

- Vim은 크게 보면 Normal/Visual/Input 모드로 나누어서 볼 수 있습니다
- 여기서 Normal mode, Visual mode 위주로 살펴보겠습니다.
- 그리고 Terminal Mode도 살펴보겠습니다.

놀랍게도, 터미널 모드라는게 있습니다

- 터미널 에뮬레이터에서 탭을 새로 띄우지 않고도 터미널을 이용할 수 있습니다.
(Thanks to Terminal mode)
- 사용자의 입력을 요구하는 터미널 명령어를 실행할때,
터미널 모드를 이용하면 다른 탭 띄우지 않고도 온전히 플로우를 유지할 수 있습니다.

터미널 모드로 할 수 있는 것들이 많습니다

- Aider (similar to Github Copilot Chat)
- Github CLI
- Dive (Docker image explorer)

굳이 필요하지도 않은 플러그인을 위한 키맵을 만들 필요도 없습니다

Normal 모드로 스크립트 실행하기

- ex) 지금 열고 있는 파일을 dotfiles 변경사항에 추가하기 -- chezmoi add %
- ex) 지금 열고 있는 erb 파일을 포매터에 넘기기 -- bundle exec htmlbeautifier %

```
-- format on save for erb
vim.api.nvim_create_autocmd("BufWritePre", {
  pattern = "*.{erb}",
  callback = function()
    -- run "bundle exec htmlbeautifier %" if htmlbeautifier is installed
    -- and update buffer immediately
    vim.cmd("write!")
    vim.cmd("silent !bundle exec htmlbeautifier %")
    vim.cmd("edit!")
  end,
})
```

Visual 모드도 굉장히 좋은데 사람들이 몰라요

- CLI 도구와 함께라면 정말 정말 강력합니다

외부 명령어에다가 선택한 라인들을 **pipe**로 맡아주기만 하면 됩니다

```
78 08 Let You, 2024-09-14 1
1 16 Desert You, 2024-09-1
2 01 Never You, 2024-09-14
3 10 Down You, 2024-09-14
4 02 Gonna You, 2024-09-14
5 03 Give You, 2024-09-14
6 04 You You, 2024-09-14 1
7 05 Up You, 2024-09-14 17
8 14 Around You, 2024-09-1
9 12 Gonna You, 2024-09-14
10 13 Run You, 2024-09-14 1
11 11 Never You, 2024-09-14
12 06 Never You, 2024-09-14
13 09 You You, 2024-09-14 1
14 07 Gonna You, 2024-09-14
15 15 And You, 2024-09-14
16 17 You You, 2024-09-14 1

'<,'>sort Variable
'<,'>stopinsert Variable -12
'<,'>sort
```

```
+ 78 01 Never You, 2024-0
+ 1 02 Gonna
+ 2 03 Give
+ 3 04 You
+ 4 05 Up
+ 5 06 Never
+ 6 07 Gonna
+ 7 08 Let
+ 8 09 You
+ 9 10 Down
+ 10 11 Never
+ 11 12 Gonna
+ 12 13 Run
+ 13 14 Around
+ 14 15 And
+ 15 16 Desert
+ 16 17 You
```

플러그인을 찾고 싶겠지만, 이미 대체재가 있어요

- **Sounds like a tautology,**
but some plugins don't exist because better alternatives already exist.
- For example, searching with fzf, ripgrep, or ast-grep can be often more efficient than using a plugin.
- Sometimes, the absence of a plugin can be a sign that **the task is better suited for existing CLI tools or built-in functionality.**

CI도 직접 짤 수 있어요

Pipe/Popen (CLI 도구를 찐다면 애부터 주목!)

- Capturing output
 - Use stdout and stderr to capture standard output and error.
 - Example: Python subprocess.Popen() and Ruby's backticks/system.
- Exporting CLI Output (json/csv/xml/yaml)
 - Many CLI tools support exporting output in structured formats like JSON.
- Processing output with your favorite language (jq, stdlib, ...)

example: Popen

```
~/quinjet  profile-page ?  @ v22.5.1  18:45
> ruby -e "puts `gh pr list --state=closed --json author,url --limit=2`"
[{"author":{"id":"MDQ6VXNlcjI0Mjc5NjM=", "is_bot":false, "login":"malkoG", "name":"Lee Jae-yeol"}, "url":"https://github.com/malkoG/quinjet/pull/8"}, {"author":{"id":"MDQ6VXNlcjI0Mjc5NjM=", "is_bot":false, "login":"malkoG", "name":"Lee Jae-yeol"}, "url":"https://github.com/malkoG/quinjet/pull/7"}]
```


example: Popen

```
~/quijet profile-page ? v22.5.1 18:45
> ruby -e "puts `gh pr list --state=closed --json author,url --limit=2`"
[{"author":{"id":"MDQ6VXNlcjI0Mjc5NjM=","is_bot":false,"login":"malkoG","name":"Lee Jae-yeol"},"url":"https://github.com/malkoG/quijet/pull/8"}, {"author":{"id":"MDQ6VXNlcjI0Mjc5NjM=","is_bot":false,"login":"malkoG","name":"Lee Jae-yeol"},"url":"https://github.com/malkoG/quijet/pull/7"}]

~/quijet profile-page ? v22.5.1 18:45
> ruby -e "puts `gh pr list --state=closed --json author,url --limit=2`" | jq
[
  {
    "author": {
      "id": "MDQ6VXNlcjI0Mjc5NjM=",
      "is_bot": false,
      "login": "malkoG",
      "name": "Lee Jae-yeol"
    },
    "url": "https://github.com/malkoG/quijet/pull/8"
  },
  {
    "author": {
      "id": "MDQ6VXNlcjI0Mjc5NjM=",
      "is_bot": false,
      "login": "malkoG",
      "name": "Lee Jae-yeol"
    },
    "url": "https://github.com/malkoG/quijet/pull/7"
  }
]
```

example: Popen

```
~/quijet  profile-page ?  v22.5.1  18:45
> ruby -e "puts `gh pr list --state=closed --json author,url --limit=2`"
[{"author":{"id":"MDQ6VXNlcjI0Mjc5NjM=","is_bot":false,"login":"malkoG","name":"Lee Jae-yeol"},"url":"https://github.com/malkoG/quijet/pull/8"}, {"author":{"id":"MDQ6VXNlcjI0Mjc5NjM=","is_bot":false,"login":"malkoG","name":"Lee Jae-yeol"},"url":"https://github.com/malkoG/quijet/pull/7"}]

~/quijet  profile-page ?  v22.5.1  18:45
> ruby -e "puts `gh pr list --state=closed --json author,url --limit=2`" | jq
[
  {
    "author": {
      "id": "MDQ6VXNlcjI0Mjc5NjM=",
      "is_bot": false,
      "login": "malkoG",
      "name": "Lee Jae-yeol"
    },
    "url": "https://github.com/malkoG/quijet/pull/8"
  },
  {
    "author": {
      "id": "MDQ6VXNlcjI0Mjc5NjM=",
      "is_bot": false,
      "login": "malkoG",
      "name": "Lee Jae-yeol"
    },
    "url": "https://github.com/malkoG/quijet/pull/7"
  }
]

~/quijet  profile-page ?  v22.5.1  18:45
> ruby -e "puts `gh pr list --state=closed --json author,url --limit=2`" | jq | ruby -rjson -e "puts JSON.parse($<.read)"
{"author"=>{"id"=>"MDQ6VXNlcjI0Mjc5NjM=", "is_bot"=>false, "login"=>"malkoG", "name"=>"Lee Jae-yeol"}, "url"=>"https://github.com/malkoG/quijet/pull/8"}
{"author"=>{"id"=>"MDQ6VXNlcjI0Mjc5NjM=", "is_bot"=>false, "login"=>"malkoG", "name"=>"Lee Jae-yeol"}, "url"=>"https://github.com/malkoG/quijet/pull/7"}]
```

Gum

- <https://github.com/charmbracelet/gum>
- Interactive UI for CLI tools (prompt, single-select, multiple-select)
- Create more user-friendly CLIs
 - Gum helps bridge the gap between terminal tools and user-friendliness, providing interactive prompts.
- Quick and Easy to integrate

Gum + (python/ruby/...) = ♥

- 여러분이 좋아하는 언어에 Gum을 감싸서 드셔보세요:
 - Python (Typer) / Ruby (Thor) / Rust (Clap)
- UI/UX는 Gum에게 맡기고, 중요한 로직은 여러분이 좋아하는 언어로

My secret weapon: exploring git diff

- 내가 보고 싶은 커밋만 보기
- 관심없는 변경사항은 필터링하고, 중요한 변경사항에만 집중하기

My secret weapon: Todoist

- Vim에다가 Todoist 통합해서 할 일 관리하기
- https://github.com/malkoG/dotfiles/blob/main/private_dot_config/mise/tasks/3rdparty/executable_todoist
- Additional functionality
 - 갤럭시 버즈 / 갤럭시 워치 / 애플 워치 / 애플 에어팟 등등을 통해서 알림 받기

Takeaway

- 여러분이 생각하는 것보다 Visual/Terminal 모드는 굉장히 좋습니다.
- 요구사항이랑 맞는 CLI 도구를 발견했다면 그것부터 써보세요.
 - CLI 명령어 조합에 스크립트 언어를 사용한다면,
좀 더 매끄럽게 워크플로우를 자동화할 수 있습니다.
 - 플러그인 대신, CLI 명령어를 조합하는게 더 나을 수 있어요
- popen으로 표준입출력을 다룰 줄 안다면,
CLI 도구를 짜는 것 자체는 어렵지 않아요. (그리고 Vim 플러그인이 필요없을지도..)

Thank you

Twitter : [kodingwarrior](#)

Discord : [kokoko.kojima](#)

Email : jaeyeol.lee@hey.com