My Vim life

VimConf 2019
@gorilla0513
About me

• **Name**
  Gorilla (Twitter: @gorilla0513)

• **Job**
  Ueno zoo (My home) guardian

• **Vim experience**
  1 year

• **Hobby**
  Developing Vim plugins and tool, and writing articles.
What I like
Banana!!! (other fruit too)
Small animals!! (cat, dog, hamster)
What I made
# docker.vim

https://github.com/skanehira/docker.vim

## Features

### images
- image list (update every 5 second)
- delete image
- pull image
- search image
- open DockerHub in browser
- push an image
- tag an image
- build an image
- save an image to tarball
- inspect an image

### containers
- container list (update every 5 second)
- start/stop/restart/kill container
- delete container
- attach container
- run container (like docker run (container))
- monitoring container logs and CPU/MEM
- copy file/folders between containers and local filesystem

### networks
- network list (update every 5 second)

### others
- monitoring docker engine’s events
docker-compose.vim

https://github.com/skanehira/docker-compose.vim

Usage

If `{file}` is not specified, the plugin will find `docker-compose.yml` or `docker-compose.yaml` from the current directory and use it.

<table>
<thead>
<tr>
<th>vim ex command</th>
<th>docker-compose command</th>
</tr>
</thead>
<tbody>
<tr>
<td>DockerCompose (args)</td>
<td>docker-compose (args)</td>
</tr>
<tr>
<td>DockerComposeList {file}</td>
<td>docker-compose -f {file} ps</td>
</tr>
<tr>
<td>DockerComposeLogs {file}</td>
<td>docker-compose -f {file} logs</td>
</tr>
<tr>
<td>DockerComposeUp {file}</td>
<td>docker-compose -f {file} up</td>
</tr>
<tr>
<td>DockerComposeDown {file}</td>
<td>docker-compose -f {file} down</td>
</tr>
<tr>
<td>DockerComposeDownAll {file}</td>
<td>docker-compose -f {file} down --rm all -v</td>
</tr>
<tr>
<td>DockerComposeStop {file}</td>
<td>docker-compose -f {file} stop</td>
</tr>
<tr>
<td>DockerComposeStart {file}</td>
<td>docker-compose -f {file} start</td>
</tr>
<tr>
<td>DockerComposeRestart {file}</td>
<td>docker-compose -f {file} restart</td>
</tr>
<tr>
<td>DockerComposeBuild {file}</td>
<td>docker-compose -f {file} build</td>
</tr>
<tr>
<td>DockerComposePull {file}</td>
<td>docker-compose -f {file} pull</td>
</tr>
<tr>
<td>DockerComposeConfig {file}</td>
<td>docker-compose -f {file} config</td>
</tr>
<tr>
<td>DockerComposeServices {file}</td>
<td>docker-compose -f {file} services</td>
</tr>
<tr>
<td>DockerComposeCreate {file}</td>
<td>docker-compose -f {file} up --no-start</td>
</tr>
<tr>
<td>DockerComposeRemove {file}</td>
<td>docker-compose -f {file} rm -f</td>
</tr>
</tbody>
</table>
translate.vim

https://github.com/skanehira/translate.vim

Usage

The default is to translate English into Japanese.

The language code is bellow. https://cloud.google.com/translate/docs/languages

Translate current line

:Translate

Translate specified words

:Translate hello my name is gorilla

Toggle between resource and target to translate when using "!"

:Translate!

Translate selected lines

:'<,'>Translate

Auto translate mode enable

:AutoTranslateModeEnable
docui (Docker TUI Client)

https://github.com/skanehira/docui

docui is a TUI Client for Docker. It can do the following:

- **image**
  - search/pull/remove
  - save/import/load
  - inspect/filtering

- **container**
  - create/remove
  - start/stop
  - export/commit
  - inspect/rename/filtering
  - exec cmd

- **volume**
  - create/remove
  - inspect/filtering

- **network**
  - remove
  - inspect/filtering
pst (TUI process monitor)
https://github.com/skanehira/pst
tson (TUI JSON viewer & editor)

https://github.com/skanehira/tson

Usage

- From file
  - $ tson < test.json

- From pipe
  - $ curl -X POST http://gorilla/likes/regist | tson

- From URL (only can use http get method)
  - $ tson -u http://gorilla/likes/json

Keybinding

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>move down</td>
</tr>
<tr>
<td>k</td>
<td>move up</td>
</tr>
<tr>
<td>g</td>
<td>move to the top</td>
</tr>
<tr>
<td>G</td>
<td>move to the bottom</td>
</tr>
<tr>
<td>ctrl-f</td>
<td>page up</td>
</tr>
<tr>
<td>ctrl-b</td>
<td>page down</td>
</tr>
<tr>
<td>h</td>
<td>hide current node</td>
</tr>
<tr>
<td>H</td>
<td>collapse all nodes</td>
</tr>
<tr>
<td>l</td>
<td>expand current node</td>
</tr>
<tr>
<td>L</td>
<td>expand all nodes</td>
</tr>
<tr>
<td>r</td>
<td>read from file</td>
</tr>
<tr>
<td>s</td>
<td>save to file</td>
</tr>
<tr>
<td>a</td>
<td>add new node</td>
</tr>
<tr>
<td>A</td>
<td>add new value</td>
</tr>
<tr>
<td>d</td>
<td>clear children nodes</td>
</tr>
<tr>
<td>Enter</td>
<td>edit node</td>
</tr>
<tr>
<td>/</td>
<td>search nodes</td>
</tr>
<tr>
<td>?</td>
<td>show helps</td>
</tr>
</tbody>
</table>
About my session

This year’s main theme is “how to be more productive with Vim?”. So, I’m going to look back at what I’ve done over the past year and talk about how I learned Vim and to be more productive with Vim.
# Simple chronology

<table>
<thead>
<tr>
<th>Years</th>
<th>What did I do</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018/09</td>
<td>Start of using Vim at work</td>
</tr>
<tr>
<td>2018/10-11</td>
<td>Using Vim every day to learn Vim</td>
</tr>
<tr>
<td>2018/12</td>
<td>Developed my first Vim plugin</td>
</tr>
<tr>
<td>2019/01</td>
<td>Developed tools with Vim</td>
</tr>
<tr>
<td>2019/02</td>
<td>Start of holding an event “gorilla.vim” every month</td>
</tr>
<tr>
<td>2019/03-04</td>
<td>Developed tools with Vim</td>
</tr>
<tr>
<td>2019/05</td>
<td>Developed Vim plugins and tools.</td>
</tr>
<tr>
<td></td>
<td>Started write an series of articles about Vim at Sakura Knowledge every month.</td>
</tr>
<tr>
<td>2019/06</td>
<td>Published a book “Vimが好きになる本”</td>
</tr>
<tr>
<td>2019/07-</td>
<td>Developed Vim plugins and tools.</td>
</tr>
<tr>
<td></td>
<td>Writing books, articles, etc… up to now.</td>
</tr>
</tbody>
</table>
2018/09
Start of using Vim at work
Why I started using Vim

The reason is that I had to work development on Linux.
I used IDE(Goland, Eclipse, VSCode) before Vim.
•How was it using Vim

Vim is very difficult to use because it is completely different from the modeless editor.
• Why I continue using Vim
  I can write code in IDE, but I still used Vim. Because I believed I could write code faster if I could get used to Vim.
2018/10~11
Learning Vim
• **Using Vim every day to do something.**
  My goal was to operate Vim without thinking.

  e.g Writing code, articles, memo, etc...
• **Learned what is capable in each modes**

   I read helps (index, starting, insert, etc...), articles and book (*Practical Vim: Edit Text at the Speed of Thought*) to learned useful functions of Vim.
• Result of learning Vim
  I got able to operate Vim as I wanted, and got faster to write code than before.
Vim, it’s awesome!!
2018/12
Developed My first Vim plugin
• **About plugin**
  The plugin name is “vsession” that can save more easily and load vim’s sessions.
Why I made plugin

I wanted to try making a Vim plugin and learn more about Vim.
• **What I thought after making a Vim plugin**

Vim plugins are not difficult to make. I didn't understand Vim script well, but I could make it with watching example and articles.
2019/02
Started “gorilla.vim”
• **About gorilla.vim**

  gorilla.vim is an event where people who are interested in Vim and Vimmer gather to perform LT and sessions.

  Approximately 40 people gather on average every month.
• Why I started holding an event
  Because I want to get knowledges of Vim and interact with Vimmer.
• **Result of holding the event**
  I got knowledges of Vim, and got able to communicate with many Vimmer.
2019/05
Started write a series of article about Vim at "Sakura Knowledge" every month
• About the "Sakura Knowledge"

"Sakura Knowledge" is an information site for IT engineers.
Why I started writing a serial article
Fortunately, I was offered a job of writing a serial article on Vim.
I accepted the offer because I want many people to know the awesomeness of Vim.
2019/06
Published a book
"Vimが好きになる本"
About the book

This book is for beginners and intermediate. Including mainly the following contents

- what can do in each modes
- how to use buffer, window and tab page
- how to use help
- how to customize vim
- introduction of useful plugins
- how to write Vim script and plugin
• Why I wrote book
  I want more people to know that Vim has a lot of useful features and using them can improve your work efficiency.
• Result of writing book
  I learned about the detailed features of Vim. Also, more than 500 books are sold 🎉
2019/07~
Developed Vim plugins and tools. writing books, articles, etc...
• Plugins made so far
  - translate.vim
  - docker.vim
  - docker-compose.vim
  - vsession
  - badapple.vim
  - say.vim
  - generatedir.vim
  - train.vim
  - weather.vim
• **Tools made so far**
  - docui
  - pst
  - tson
  - gjo
  - gocui-component
  - gol
  - gtran
  - go-enc
  - go-logger
  - go-run
  - got
• Write articles about Vim
My blog have only Vim articles.

https://gorilla.netlify.com/
What I thought of using Vim every day for a year
Vim is an awesome editor!!
• **Awesome points**
  - Can edit text faster using count, operator, motion, text objects and repeat.
  - Can format text, run example code from buffer, run curl and edit response, etc such as shell commands.
  - Can use terminal in Vim.
  - Extend features with plugins.
How learn Vim (for beginners)

1. Learn basic Vim operations (vimtutor).
2. Learn operator, motion, text objects, count and repeat.
3. Learn what can do in each mode.
4. Read help, articles, books and practice.
5. Write articles about Vim.
6. Join the Vim community and communicate with Vimmer.
7. Make a plugin.
How to be more productive with Vim
Use plugins

• Plugins make it easy to increase productivity e.g fuzzy finder, auto completion, formatter, etc...

• But I don't think it's good to rely on plugins too much
Because if we rely on plugins too much, we might not be able to do anything without them.
Use plugins

- Points to determine whether to use plugins or not

Use plugins when it is difficult to cover Vim standard features.
Use plugins

• Some of the plugins I usually use

  • junegunn/fzf.vim
    fuzzy finder can search and use some resources. e.g files, buffers, lines, etc...

  • prabirshrestha/vim-lsp
    Async LSP plugin for vim8 and neovim.
    Can auto completion, snippets and Folding.

  • thinca/vim-quickrun
    Can execute whole/part of editing file.

  • SirVer/ultisnips
    The ultimate snippet solution for Vim.
Use plugins

• Some of the plugins I usually use

  • mattn/sonictemplate-vim
    Can speed up easily your coding method.

  • skanehira/translate.vim(my plugin)
    Can translate language with Google translate.

  • skanehira/docker.vim(my plugin)
    Can manage docker resources on Vim.

  • skanehira/docker-compose.vim(my plugin)
    Can manage docker-compose resources on Vim.
Improve vimrc

• Improve vimrc to match own usage and increase work efficiency

even without plugins, there are things that can be improved by writing in vimrc.

e.g. display line number, highlight search, restore cursor, auto indent, mapping keys, display whitespace at the end of the line in red, etc…
Make plugin

• By making a plugin, we will become familiar with the Vim specification

Through the development of the plugin, I learned about extension of functions using external commands, buffer and window operations, etc... in Vim.
Keep learning Vim

Vim is improving daily and will evolve.

Useful features will be added, and existing features will be improved.

I recommend you to check GitHub regularly.
Summary

• Vim is awesome editor.

• Good ways to increase productivity with Vim
  - Use plugins
  - Improve vimrc
  - Create plugins
  - Continue learning Vim

• But, there is no need to use Vim forcibly. The important thing is whether you can enjoy Vim or not.
Special thanks

• Those who taught me a lot about Vim
  mattn, KoRoN, thinca, Tsuyoshi CHO, rbtnn, kuu and other vim-jp members

• Gorilla.vim organizers
  daisuzu, marcy, shinko, hoda

• Everyone who participated in Gorilla.vim

• VimConf staff

• Those who contribute to Vim

• Those who gave an English review of the slide
  ujihisa, murakami kouhei
Thank you