SpaceVim

A modular Vim/Neovim configuration

Development

SpaceVim is a joint effort of all contributors. We encourage you to participate in SpaceVim's development. This page describes the entire development process of SpaceVim.

We have some guidelines that we need all contributors to follow. You can only think about reading the part that is relevant to what you are going to do:

Contributing code

The source code of spacevim is hosted at github. Code and documentation contributions of any kind are welcome.

License

The license is GPLv3 for all the parts of SpaceVim. This includes:

- The initialization and core files.
- All the layer files.
- The documentation

For files not belonging to SpaceVim like bundle packages, refer to the header file. Those files should not have an empty header, we may not accept code without a proper header file.

Conventions

SpaceVim is based on conventions, mainly for naming functions, keybindings definition and writing documentation. Please read these conventions to make sure you understand them before you contribute code or documentation for the first time.

Commit style guide

Follow the conventional commits guidelines to *make reviews easier* and to make the git logs more valuable. The general structure of a commit message is:

```
<type>([optional scope]): <description>

[optional body]

[optional footer(s)]
```

types:

- `feat`: A new feature
- `fix`: A bug fix
- `docs`: Documentation only changes
- `style`: Changes that do not affect the meaning of the code
- `refactor`: A code change that neither fixes a bug nor adds a feature
- `pref`: A code change that improves performance
- `test`: Adding missing tests or correcting existing tests
- `ci`: Changes to our CI configuration files and scripts
- `chore`: Other changes that don’t modify src or test files
- `revert`: Reverts a previous commit

scopes:

- `api`: files in `autoload/SpaceVim/api/` and `docs/api/` directory
- `layer`: files in `autoload/SpaceVim/layers/` and `docs/layers/` directory
- `plugin`: files in `autoload/SpaceVim/plugins/` directory
- `bundle`: files in `bundle/` directory
- `core`: other files in this repository

In addition to these scopes above, you can also use a specific layer name or plugin name as a scope.

subject:

Subjects should be no greater than 50 characters, should not begin with a capital letter and do not end with a period.

Use an imperative tone to describe what a commit does, rather than what it did. For example, use change; not changed or changes.

body:

Not all commits are complex enough to warrant a body, therefore it is optional and only used when a commit requires a bit of explanation and context.

footer

The footer is optional and is used to reference issue tracker IDs.

Breaking change

Breaking changes must be indicated by “!” after the type/scope, and a “BREAKING CHANGE” footer describing the change. Example:

```
refactor(tools#mpv)!: change default musics_directory

BREAKING CHANGE: `~/Music` is standard on macOS and
also on FreeDesktop's XDG.
```

Contributing a layer

Please read the layers documentation first.

Layer with no associated configuration will be rejected. For instance a layer with just a package and a hook can be easily replaced by the usage of the variable `custom_plugins`.

File header

The file header for Vim script should look like the following template:

```
"=============================================================================
" FILENAME --- NAME layer file for SpaceVim
" Copyright (c) 2016-2023 Wang Shidong & Contributors
" Author: Wang Shidong < wsdjeg@outlook.com >
" URL: https://spacevim.org
" License: GPLv3
"=============================================================================
```

You should replace FILENAME by the name of the file (e.g. foo.vim) and NAME by the name of the layer you are creating, don’t forget to replace **YOUR NAME** and **YOUR EMAIL** too.

Author of a new layer

In the file’s header, replace the default author name (Shidong Wang) with your name.

The following example shows how to create a new layer named `foo`:

1. Fork SpaceVim repo.
2. Add a layer file `autoload/SpaceVim/layers/foo.vim` for `foo` layer.

3. Edit layer file, check out the example below:

```
"==============================================================================
" foo.vim --- foo Layer file for SpaceVim
" Copyright (c) 2012-2022 Shidong Wang & Contributors
" Author: Shidong Wang < wsdjeg@outlook.com >
" URL: https://spacevim.org
" License: GPLv3
"==============================================================================


""
" @section foo, layers-foo
" @parentsection layers
" This is the doc for this layer:
"
" @subsection Key Bindings
" >
"   Mode      Key           Function
"   -------------------------------------------------------------
"   normal    <leader>jA    generate accessors
"   normal    <leader>js    generate setter accessor
" <
" @subsection Layer options
" >
"   Name              Description                         Default
"   -------------------------------------------------------------
"   option1       Set option1 for foo layer                 ''
"   option2       Set option2 for foo layer                 []
"   option3       Set option3 for foo layer                 {}
" <
" @subsection Global options
" >
"   Name              Description                         Default
"   -------------------------------------------------------------
"   g:pluginA_opt1    Set opt1 for plugin A                 ''
"   g:pluginB_opt2    Set opt2 for plugin B                 []
" <

function! SpaceVim#layers#foo#plugins() abort
  let plugins = []
  call add(plugins, ['Shougo/foo.vim', {'option' : 'value'}])
  call add(plugins, ['Shougo/foo_test.vim', {'option' : 'value'}])
  return plugins
endfunction
```

```

function! SpaceVim#layers#foo#config() abort
  let g:foo_option1 = get(g:, 'foo_option1', 1)
  let g:foo_option2 = get(g:, 'foo_option2', 2)
  let g:foo_option3 = get(g:, 'foo_option3', 3)
  " ...
endfunction

" add layer options:
let s:layer_option = 'default var'
function! SpaceVim#layers#foo#set_variable(var) abort
  let s:layer_option = get(a:var, 'layer_option', s:layer_option)
endfunction

" completion function for layer options:
function! SpaceVim#layers#foo#get_options() abort
    return ['layer_option']
endfunction
```

1. Create the layer's documentation file `docs/layers/foo.md` for `foo` layer.
2. Open `docs/layers/index.md`, and run `:call SpaceVim#dev#layers#update()` to update the layers list.
3. Send a PR to SpaceVim.

Contributor to an existing layer

If you want to contribute to an already existing layer, you should not modify any header file.

Contributing a keybinding

Mappings are an important part of SpaceVim.

First if you want to have some personal mappings. This can be done in your bootstrap function.

If you think it is worth contributing new mappings, be sure to read the documentation to find the best mappings, then create a Pull-Request with your mappings.

ALWAYS document your new mappings or mapping changes inside the relevant documentation file. It should be the layername.md and the documentation.

Language specified key bindings

All language specified key bindings have the prefix `SPC l`.

We recommend you to use the common language specified key bindings for the same purpose as the following:

Key Binding	Description
`g d`	jump to definition

Key Binding	Description
`g D`	jump to type definition
`SPC l r`	start a runner for current file
`SPC l e`	rename symbol
`SPC l d`	show doc
`K`	show doc
`SPC l i r`	remove unused imports
`SPC l i s`	sort imports with isort
`SPC l s i`	Start a language specified inferior REPL process
`SPC l s b`	send buffer and keep code buffer focused
`SPC l s l`	send line and keep code buffer focused
`SPC l s s`	send selection text and keep code buffer focused

All above key bindings are just recommended as default, but they are also based on the language layer itself.

Contributing a banner

The startup banner is the SpaceVim logo by default. but there are also ASCII banners available in the core/banner layer.

If you have some ASCII skills you can submit your artwork!

You are free to choose a reasonable height size. but the width size should be around 75 characters.

Bundle plugins

In `bundle/` directory, there are two kinds of plugins:

1. unmodified plugins, same as the upstream.
2. modified plugins based on specific commit.

checkout the bundle plugins page for more info.

Build with SpaceVim

SpaceVim provides a lot of public APIs, you can create plugins based on these APIs. Also you can add a badge to the README.md of your plugin.

build with SpaceVim

markdown

```
[![](https://spacevim.org/img/build-with-SpaceVim.svg)](https://spacevim.org)
```

Newsletters

- ### Newsletter #2 - Never lost, Never give up

  01 Jan 2018
  We know exactly what is the purpose of SpaceVim, and we keep trying to get it, never get lost, never give up...

- ### Newsletter #1 - A New Hope

  31 May 2017
  A new hope: turn vim/neovim to be an IDE for most languages

Changelog

- ### SpaceVim release v2.4.0

  22 Dec 2024
  SpaceVim releases v2.4.0, which optimizes Neovim support and rewrites the core plugin using Lua

- ### SpaceVim release v2.3.0

  23 Mar 2024
  SpaceVim release v2.3.0 with more lua plugins and better experience.

- ### SpaceVim release v2.2.0

  05 Jul 2023
  SpaceVim release v2.2.0 with more lua plugins and better experience.

- ### SpaceVim release v2.1.0

  30 Mar 2023
  SpaceVim release v2.1.0 with more lua plugins and better experience.

- ### SpaceVim release v2.0.0

  02 Jul 2022
  The second major release of SpaceVim adds several layers and lua plugins for a better use experience

- ### SpaceVim release v1.9.0

  06 Jan 2022
  SpaceVim release v1.9.0 with new features and better experience.

- ## SpaceVim release v1.8.0

  04 Oct 2021
  SpaceVim release v1.8.0 with lua plugins enabled and better experience.

- ## SpaceVim release v1.7.0

  14 Jun 2021
  SpaceVim release v1.7.0 with four with a number of language layers and new features.

- ## SpaceVim release v1.6.0

  31 Dec 2020
  SpaceVim release v1.6.0 with four new language layers and floating window support.

- ## SpaceVim release v1.5.0

  01 Aug 2020
  SpaceVim release v1.5.0 with four new language layers and floating window support.

- ## SpaceVim release v1.4.0

  05 Apr 2020
  SpaceVim released V1.4.0 with eight new language layers and an asynchronous task management plugin

- ## SpaceVim release v1.3.0

  04 Nov 2019
  SpaceVim release v1.3.0 with ten new language layers and several functional improvements

- ## SpaceVim release v1.2.0

  17 Jul 2019
  SpaceVim release v1.2.0 with 12 new programming language layers and ton of bug fixs and new features.

- ## SpaceVim release v1.1.0

  08 Apr 2019
  SpaceVim v1.1.0 introduced new floating window mapping guide, and a disk management for Windows, as well as improved the fzf layer.

- ## SpaceVim release v1.0.0

  25 Dec 2018
  V1.0.0 is the first stable version of SpaceVim, which is mainly about experience and user documentation.

- ## SpaceVim release v0.9.0

  26 Sep 2018

  SpaceVim release v0.9.0 with 15+ new language layers and frequency support for colorscheme layer.

- ## SpaceVim release v0.8.0

  18 Jun 2018

  Use toml as default configuration file for SpaceVim

- ## SpaceVim release v0.7.0

  18 Mar 2018

  Highlight symbol mode and iedit mode come out with v0.7.0

- ## SpaceVim release v0.6.0

  30 Dec 2017

  Many new features come out with v0.6.0, including c/cpp suport and code runner

- ## SpaceVim release v0.5.0

  06 Nov 2017

  SpaceVim v0.5.0 comes out with five new language layer and many new features

- ## SpaceVim release v0.4.0

  05 Aug 2017

  New release v0.4.0 provides a new UI for plugin manager and built-in flygrep plugin

- ## SpaceVim release v0.3.1

  27 Jun 2017

  New release v0.3.1 provides better key binding guide for buffer and file, including z and g prefix guide.

- ## SpaceVim release v0.3.0

  31 May 2017

  The major change happend in v0.3.0 is new mapping guide and custom mapping prefix SPC.

- ## SpaceVim release v0.2.0

  30 Mar 2017

  Mnemonic key bindings in SpaceVim

- ## SpaceVim release v0.1.0

  26 Jan 2017
  First public release of SpaceVim with IDE-like UI, most language support, modular configuration

Powered by Jekyll