



SpaceVim

A community-driven vim distribution

[主页](#) | [关于我们](#) | [使用文档](#) | [开发指南](#) | [用户社区](#) | [赞助](#)

使用文档

- » [核心思想](#)
- » [显著特性](#)
- » [运行截图](#)
- » [基本概念](#)
- » [适用人群](#)
- » [更新回滚](#)
 - » [自身更新](#)
 - » [更新插件](#)
 - » [获取日志](#)
- » [用户配置](#)
 - » [启动函数](#)
 - » [Vim 兼容模式](#)

[OPEN CHAT](#)

- » 私有模块
- » 调试上游插件
- » 界面元素
 - » 颜色主题
 - » 字体
 - » 界面元素切换
 - » 状态栏
 - » 标签栏
- » 基本操作
 - » 窗口管理器
 - » 文件操作
 - » 编辑器界面
 - » 原生功能
 - » 标签管理
 - » 模糊搜索
 - » 交互
 - » 快捷键
 - » 获取帮助信息
 - » 可用模块
 - » 界面元素显示切换
- » 常规操作
 - » 光标移动
 - » 使用 vim-easymotion 快速跳转
 - » 快速跳到网址 (TODO)
 - » 常用的成对快捷键
 - » 跳转，合并，拆分
 - » 跳转
 - » 合并，拆分
 - » 窗口操作
 - » 窗口操作常用快捷键

[OPEN CHAT](#)

- » 文件和缓冲区操作
 - » 缓冲区操作
 - » 新建空白 buffer
 - » 特殊 buffer
 - » 文件操作相关快捷键
 - » Vim 和 SpaceVim 相关文件
- » 文件树
 - » 文件树中的常用操作
 - » 文件树中打开文件
- » 以 g 为前缀的快捷键
- » 以 z 开头的命令
- » 搜索
 - » 使用额外工具
 - » 配置搜索工具
 - » 常用按键绑定
 - » 在当前文件中进行搜索
 - » 搜索当前文件所在的文件夹
 - » 在所有打开的缓冲区中进行搜索
 - » 在任意目录中进行搜索
 - » 在工程中进行搜索
 - » 后台进行工程搜索
 - » 在网上进行搜索
 - » 实时代码检索
 - » 保持高亮
 - » 高亮光标下变量
- » 编辑
 - » 粘贴文本
 - » 粘贴文本自动缩进
 - » 文本操作命令
 - » 文本插入命令
 - » 增加或减小数字

[OPEN CHAT](#)

- » [ledit 多光标编辑](#)
 - » [ledit 快捷键](#)
- » [注释 \(Commentings\)](#)
- » [多方式编码](#)
- » [异步运行器和交互式编程](#)
- » [错误处理](#)
- » [工程管理](#)
 - » [在工程中搜索文件](#)
- » [格式规范](#)
- » [Vim 服务](#)
- » [Achievements](#)
 - » [issues](#)
 - » [Stars, forks and watchers](#)

核心思想

四大核心思想：记忆辅助、可视化交互、一致性、社区驱动。

如果违背了以上四大核心思想，我们将会尽力修复。

记忆辅助

所有快捷键，根据其功能的不同分为不同的组，以相应的按键作为前缀，例如 **b** 为 buffer 相关快捷键前缀，**p** 为 project 相关快捷键前缀，**s** 为 search 相关快捷键前缀，**h** 为 help 相关快捷键前缀。

可视化交互

创新的实时快捷键辅助系统，以及查询系统，方便快捷查询到可用的模块、插件以及其它更多信息。

一致性

相似的功能使用同样的快捷键，这在 SpaceVim 中随处可见。这得益于明确的约定。其它模块的文档都以此为基础。

社区驱动

社区驱动，保证了 bug 修复的速度，以及新特性更新的速度。

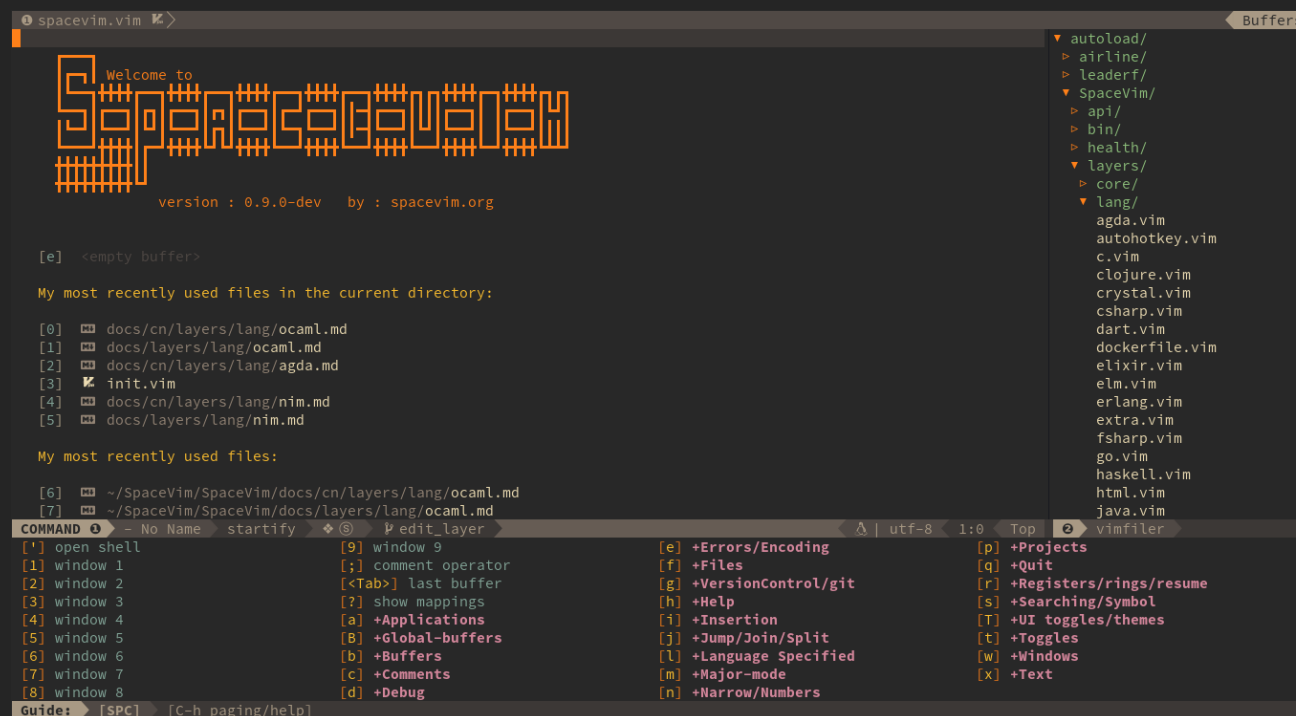
显著特性

[OPEN CHAT](#)

- » 详细的文档：在 SpaceVim 中通过 `:h SpaceVim` 来访问 SpaceVim 帮助文档。
- » 优雅简洁的界面：你将会喜欢这样的优雅而实用的界面。
- » 确保手指不离开主键盘区域：使用 Space 作为前缀键，合理组织快捷键，确保手指不离开主键盘区域。
- » 快捷键辅助系统：SpaceVim 所有快捷键无需记忆，当输入出现停顿，会实时提示可用按键及其功能。
- » 更快的启动时间：得益于 dein.vim, SpaceVim 中 90% 的插件都是按需载入的。
- » 更少的肌肉损伤：频繁使用空格键，取代 `ctrl`, `shift` 等按键，大大减少了手指的肌肉损伤。
- » 更易扩展：依照一些约定，很容易将现有的插件集成到 SpaceVim 中来。
- » 完美支持 **Neovim**: 依赖于 Neovim 的 `remote` 插件以及异步 API，SpaceVim 运行在 Neovim 下将更加完美的体验。

运行截图

欢迎页面



```
spacevim.vim >

Welcome to
SpaceVim
version : 0.9.0-dev by : spacevim.org

[e] <empty buffers>

My most recently used files in the current directory:

[0] docs/cn/layers/lang/ocaml.md
[1] docs/layers/lang/ocaml.md
[2] docs/cn/layers/lang/agda.md
[3] init.vim
[4] docs/cn/layers/lang/nim.md
[5] docs/layers/lang/nim.md

My most recently used files:

[6] ~/SpaceVim/SpaceVim/docs/cn/layers/lang/ocaml.md
[7] ~/SpaceVim/SpaceVim/docs/layers/lang/ocaml.md

COMMAND - No Name startify edit_layer
['] open shell
[1] window 1
[2] window 2
[3] window 3
[4] window 4
[5] window 5
[6] window 6
[7] window 7
[8] window 8
[9] window 9
[;] comment operator
[<Tab>] last buffer
[?] show mappings
[a] +Applications
[B] +Global-buffers
[b] +Buffers
[c] +Comments
[d] +Debug
[e] +Errors/Encoding
[f] +Files
[g] +VersionControl/git
[h] +Help
[i] +Insertion
[j] +Jump/Join/Split
[l] +Language Specified
[m] +Major-mode
[n] +Narrow/Numbers
[p] +Projects
[q] +Quit
[r] +Registers/rings/resume
[s] +Searching/Symbol
[t] +Toggles/themes
[w] +Windows
[X] +Text

Guide: [SPC] [C-h paging/help]
```

OPEN CHAT

工作界面



Neovim 运行在 iTerm2 上，采用 SpaceVim，配色为：*base16-solarized-dark*

展示了一个通用的前端开发界面，用于开发：JavaScript (jQuery), SASS, 和 PHP buffers。

图中包含了一个 Neovim 的终端，一个语法树窗口，一个文件树窗口以及一个 TernJS 定义窗口

想要查阅更多截图，请阅读 [issue #415](#)

基本概念

临时快捷键菜单

SpaceVim 根据需要定义了很多临时快捷键，这可以避免需要重复某些操作时过多按下 SPC 前缀键。当临时快捷键启用时，会在窗口下方打开一个快捷键介绍窗口，提示每一临时快捷键的功能。此外一些额外的

OPEN CHAT

辅助信息也将会显示出来。

文本移动临时快捷键：

```
custom key in sign column and error feedback for checker.  
* 73.1k documentation.md markdown ↩ ↲ manipulation unix | utf-8 308:1 15%  
Move Text Transient State  
[J] move text down [K] move text up  
[KEY] exits state [KEY] will not exit  
Transient State
```

适用人群

- » 初级 Vim 用户
- » 追求优雅界面的 Vim 用户
- » 追求更少肌肉损伤的 Vim 用户
- » 想要学习一种不一样的编辑文件方式的 Vim 用户
- » 追求简单但是可高度配置系统的 Vim 用户

更新回滚

自身更新

可通过很多种方式来更新 SpaceVim 的核心文件。建议在更新 SpaceVim 之前，更新一下所有的插件。具体内容如下：

自动更新

注意：默认，这一特性是禁用的，因为自动更新将会增加 SpaceVim 的启动时间，影响用户体验。如果你需要这一特性，可以将如下加入到用户配置文件中：`automatic_update = true`。

启用这一特性后，SpaceVim 将会在每次启动时候检测是否有新版本。更新后需重启 SpaceVim。

通过插件管理器更新

使用 `:SPUpdate SpaceVim` 这一命令，将会打开 SpaceVim 的插件管理器，更新 SpaceVim，具体进度会在插件管理器 buffer 中展示。

通过 git 进行更新

可通过在 SpaceVim 目录中手动执行 `git pull`，SpaceVim 在 Windows 下的默认目录为 `~/vimfilers`，但在 Linux 下则可使用如下命令：`git -C ~/.SpaceVim pull`

更新插件

OPEN CHAT

使用 `:SPUpdate` 这一命令将会更新所有插件，包括 SpaceVim 自身。当然这一命令也支持参数，参数为插件名称，可同时添加多个插件名称作为参数，同时可以使用 Tab 键来补全插件名称。

获取日志

使用 `:SPDebugInfo!` 这一命令可以获取 SpaceVim 运行时日志，同时，可以使用 `SPC h I` 使用打开问题模板。可在这个模板中编辑问题，并提交。

用户配置

初次启动 SpaceVim 时，他将提供选择目录，用户需要选择合适自己的配置模板。此时，SpaceVim 将自动在 HOME 目录生成 `~/.SpaceVim.d/init.toml`。所有用户脚本可以存储在 `~/.SpaceVim.d/`，这一文件夹将被加入 Vim 的运行时路径 `&runtimepath`。详情请阅读 `:h rtp`。

当然，你也可以通过 `SPACEVIMDIR` 这一环境变量，指定用户配置目录。当然也可以通过软链接来改变目录位置，以便配置备份。

SpaceVim 同时还支持项目本地配置，配置初始文件为，当前目录下的 `.SpaceVim.d/init.toml` 文件。同时当前目录下的 `.SpaceVim.d/` 也将被加入到 Vim 运行时路径。

所有的 SpaceVim 选项可以使用 `:h SpaceVim-config` 来查看。选项名称为原先 Vim 脚本中使用的变量名称去除 `g:spacevim_` 前缀。

完整的内置文档可以通过 `:h SpaceVim` 进行查阅。也可以通过按键 `SPC h SPC` 模糊搜索，该快捷键需要载入一个模糊搜索模块。

添加自定义插件

如果你需要添加 github 上的插件，只需要在 SpaceVim 配置文件中添加 `[[custom_plugins]]` 片段：

```
[[custom_plugins]]
  name = "lilydjwg/colorizer"
  on_cmd = ["ColorHighlight", "ColorToggle"]
  merged = false
```

以上这段配置，添加了插件 `lilydjwg/colorizer`，并且，通过 `on_cmd` 这一选项使得这个插件延迟加载。该插件会在第一次执行 `ColorHighlight` 或者 `ColorToggle` 命令时被加载。除了 `on_cmd` 以外，还有一些其它的选项，可以通过 `:h dein-options` 查阅。

禁用插件

OPEN CHAT

SpaceVim 默认安装了一些插件，如果需要禁用某个插件，可以通过 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中的 `disabled_plugins` 这一选项来操作：

```
[options]
# 请注意，该值为一个 List，每一个选项为插件的名称，而非 github 仓库地址。
disabled_plugins = ["clighter", "clighter8"]
```

启动函数

由于 toml 配置的局限性，SpaceVim 提供了两种启动函数 `bootstrap_before` 和 `bootstrap_after`，在该函数内可以使用 Vim script。可通过 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中的这两个选项 `bootstrap_before` 和 `bootstrap_after` 来指定函数名称，例如：

```
[options]
bootstrap_before = "myspacevim#before"
bootstrap_after  = "myspacevim#after"
```

启动函数文件应放置在 Vim &runtimepath 的 autoload 文件夹内。例如：

文件名：`~/.SpaceVim.d/autoload/myspacevim.vim`

```
function! myspacevim#before() abort
    let g:neomake_enabled_c_makers = ['clang']
    noremap jk <esc>
endfunction

function! myspacevim#after() abort
    iunmap jk
endfunction
```

函数 `bootstrap_before` 将在读取用户配置后执行，而函数 `bootstrap_after` 将在 VimEnter autocmd 之后执行。

如果你需要添加自定义以 `SPC` 为前缀的快捷键，你需要使用 bootstrap function，在其中加入：

OPEN CHAT

```
function! myspacevim#before() abort
    call SpaceVim#custom#SPCGroupName(['G'], '+TestGroup')
    call SpaceVim#custom#SPC('nore', ['G', 't'], 'echom 1', 'echomessage 1', 1)
endfunction
```

Vim 兼容模式

以下为 SpaceVim 中与 Vim 默认情况下的一些差异。

- » Normal 模式下 **s** 按键不再删除光标下的字符，在 SpaceVim 中，它是窗口相关快捷键的前缀（可以在配置文件中设置成其它按键）。如果希望恢复 **s** 按键原先的功能，可以通过 `windows_leader = ""` 将窗口前缀键设为空字符串来禁用这一功能。
- » Normal 模式下，按键在 Vim 默认情况下是重复上一次的 **f**、**F**、**t** 和 **T** 按键，但在 SpaceVim 中默认被用作为语言专用的前缀键。如果需要禁用此选项，可设置 `enable_language_specific_leader = false`。
- » Normal 模式下 **q** 按键在 SpaceVim 中被设置为了智能关闭窗口，即大多数情况下按下 **q** 键即可关闭当前窗口。可以通过 `windows_smartclose = ""` 使用一个空字符串来禁用这一功能，或修改为其它按键。
- » 命令行模式下 **Ctrl-a** 按键在 SpaceVim 中被修改为了移动光标至命令行行首。
- » 命令行模式下 **Ctrl-b** 按键被映射为方向键 **<Left>**，用以向左移动光标。
- » 命令行模式下 **Ctrl-f** 按键被映射为方向键 **<Right>**，用以向右移动光标。

可以通过设置 `vimcompatible = true` 来启用 Vim 兼容模式，而在兼容模式下，以上所有差异将不存在。当然，也可通过对应的选项禁用某一个差异。例如，恢复逗号 **,** 的原始功能，可以通过禁用语言专用的前缀键：

```
[options]
enable_language_specific_leader = false
```

如果发现有其它区别，可以[提交 PR](#)。

私有模块

这一部分简单介绍了模块的组成，更多关于新建模块的内容可以阅读 SpaceVim 的[模块首页](#)。

目的

OPEN CHAT

使用模块的方式来组织和管理插件，将相关功能的插件组织成一个模块，启用/禁用效率更加高。同时也节省了很多寻找插件和配置插件的时间。

结构

在 SpaceVim 中，一个模块是一个单个的 Vim 文件，例如，`autocomplete` 模块存储在 `autoload/SpaceVim/layers/autocomplete.vim`，在这个文件内有以下几个公共函数：

- » `SpaceVim#layers#autocomplete#plugins()`: 返回该模块插件列表
- » `SpaceVim#layers#autocomplete#config()`: 模块相关设置
- » `SpaceVim#layers#autocomplete#set_variable()`: 模块选项设置函数

调试上游插件

当发现某个内置上游插件存在问题，需要修改并调试上游插件时，可以依照以下步骤操作：

1. 禁用内置上游插件 比如，调试内置语法检查插件 `neomake.vim`

```
[options]
  disabled_plugins = ["neomake.vim"]
```

1. 添加自己 fork 的插件 修改配置文件 `init.toml`，加入以下部分，来添加自己 fork 的版本：

```
[[custom_plugins]]
  name = 'wsdjeg/neomake.vim'
  # note: you need to disable merged feature
  merged = false
```

或者添加本地克隆版本 使用 `bootstrap_before` 函数来添加本地路径：

```
function! myspacevim#before() abort
  set rtp+=~/path/to/your/localplugin
endfunction
```

界面元素

SpaceVim 集成了多种实用的 UI 插件，如常用的文件树、语法树等插件，配色主题默认采用的是 `gruvbox`。

OPEN CHAT

颜色主题

默认的颜色主题采用的是 `gruvbox`。这一主题有深色和浅色两种。关于这一主题一些详细的配置可以阅读 `:h gruvbox`。

如果需要修改 SpaceVim 的主题，可以在 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中修改 `colorscheme` 选项。例如，使用 Vim 自带的内置主题 `desert`：

```
[options]
  colorscheme = "desert"
  colorscheme_bg = "dark"
```

快捷键	功能描述
<code>SPC T n</code>	切换至下一个随机主题
<code>SPC T s</code>	通过 Unite 选择主题

可以在[主题模块](#)中查看 SpaceVim 支持的所有主题。

注意：

SpaceVim 在终端下默认使用了真色，因此使用之前需要确认下你的终端是否支持真色。可以阅读 [Colours in terminal](#) 了解根多关于真色的信息。

如果你的终端不支持真色，可以在 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中禁用真色支持：

```
[options]
  enable_guicolors = false
```

字体

在 SpaceVim 中默认的字体是 `SauceCodePro Nerd Font Mono`。如果你也喜欢这一字体，建议将这一字体安装到系统中。如果需要修改 SpaceVim 的字体，可以在 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中修改选项 `guifont`，默认值为：

```
[options]
  guifont = "SauceCodePro Nerd Font Mono:h11"
```

OPEN CHAT

如果指定的字体不存在，将会使用系统默认的字体，此外，这一选项在终端下是无效的，终端下修改字体，需要修改终端自身配置。

界面元素切换

大多数界面元素可以通过快捷键来隐藏或者显示（这一组快捷键以 **t** 和 **T** 开头）：

快捷键	功能描述
SPC t 8	高亮所有超过 80 列的字符
SPC t f	高亮临界列，默认 <code>max_column</code> 是第 120 列
SPC t h h	高亮当前行
SPC t h i	高亮代码对齐线
SPC t h c	高亮光标所在列
SPC t h s	启用/禁用语法高亮
SPC t i	切换显示当前对齐(TODO)
SPC t n	显示/隐藏行号
SPC t b	切换背景色
SPC t t	打开 Tab 管理器
SPC T ~	显示/隐藏 Buffer 结尾空行行首的 ~
SPC T F	切换全屏(TODO)
SPC T f	显示/隐藏 Vim 边框(GUI)
SPC T m	显示/隐藏菜单栏
SPC T t	显示/隐藏工具栏

状态栏

OPEN CHAT

`core#statusline` 模块提供了一个高度定制的状态栏，提供如下特性，这一模块的灵感来自于 spacemacs 的状态栏。

- » 展示窗口序号
- » 通过不同颜色展示当前模式
- » 展示搜索结果序号
- » 显示/隐藏语法检查信息
- » 显示/隐藏电池信息
- » 显示/隐藏 SpaceVim 功能启用状态
- » 显示版本控制信息（需要 `git` 和 `VersionControl` 模块）

快捷键	功能描述
<code>SPC [1-9]</code>	跳至指定序号的窗口

默认主题 `gruvbox` 的状态栏颜色和模式对照表：

模式	颜色
Normal	灰色
Insert	蓝色
Visual	橙色
Replace	浅绿色

以上的这几种模式所对应的颜色取决于不同的主题模式。

一些状态栏元素可以进行动态的切换：

快捷键	功能描述
<code>SPC t m b</code>	显示/隐藏电池状态 (需要安装 <code>acpi</code>)
<code>SPC t m c</code>	toggle the org task clock (available in org layer)(TODO)
<code>SPC t m i</code>	显示/隐藏输入法
<code>SPC t m m</code>	显示/隐藏 SpaceVim 已启用功能

OPEN CHAT

快捷键	功能描述
<code>SPC t m M</code>	显示/隐藏文件类型
<code>SPC t m n</code>	toggle the cat! (if colors layer is declared in your dotfile)(TODO)
<code>SPC t m p</code>	显示/隐藏光标位置信息
<code>SPC t m t</code>	显示/隐藏时间
<code>SPC t m d</code>	显示/隐藏日期
<code>SPC t m T</code>	显示/隐藏状态栏
<code>SPC t m v</code>	显示/隐藏版本控制信息

nerd 字体安装：

SpaceVim 默认使用 `nerd fonts`，可参阅其安装指南进行安装。

语法检查信息：

状态栏中语法检查信息元素如果被启用了，当语法检查结束后，会在状态栏中展示当前语法错误和警告的数量。

搜索结果信息：

当使用 `/` 或 `?` 进行搜索时，或当按下 `n` 或 `N` 后，搜索结果序号将被展示在状态栏中，使用类似于 `20/22` 这样的分数显示搜索结果的当前序号以及结果总数。具体的效果图如下：



电池状态信息：

`acpi` 可展示电池电量剩余百分比。

使用不同颜色展示不同的电池状态：

电池状态	颜色
75% - 100%	绿色
30% - 75%	黄色

OPEN CHAT

电池状态	颜色
0% - 30%	红色

所有的颜色都取决于不同的主题。

状态栏分割符：

可通过使用 `statusline_separator` 来定制状态栏分割符，例如使用常用的方向箭头作为状态栏分割符：

```
statusline_separator = 'arrow'
```

SpaceVim 所支持的分割符以及截图如下：

分割符	截图
arrow	
curve	
slant	
nil	
fire	

SpaceVim 功能模块：

功能模块可以通过 `SPC t m m` 快捷键显示或者隐藏。默认使用 Unicode 字符，可通过设置 `statusline_unicode_symbols = false` 来启用 ASCII 字符。(或许在终端中无法设置合适的字体时，可使用这一选项)。

状态栏中功能模块内的字符显示与否，同如下快捷键功能保持一致：

快捷键	Unicode	ASCII	功能
<code>SPC t 8</code>	Ⓔ	8	高亮指定列后所有字符
<code>SPC t f</code>	Ⓕ	f	高亮指定列字符

OPEN CHAT

快捷键	Unicode	ASCII	功能
SPC t s	Ⓢ	s	语法检查
SPC t S	Ⓢ	S	拼写检查
SPC t w	Ⓜ	w	行尾空格检查

状态栏的颜色

SpaceVim 默认为 **colorcheme** 模块所包含的主题颜色提供了状态栏主题，若需要使用其它颜色主题，需要自行设置状态栏主题。若未设置，则使用 gruvbox 的主题。

可以参考以下模板来设置：

```
" the theme colors should be
" [
"   \ [ a_guifg, a_guibg, a_ctermfg, a_ctermbg],
"   \ [ b_guifg, b_guibg, b_ctermfg, b_ctermbg],
"   \ [ c_guifg, c_guibg, c_ctermfg, c_ctermbg],
"   \ [ z_guibg, z_ctermbg],
"   \ [ i_guifg, i_guibg, i_ctermfg, i_ctermbg],
"   \ [ v_guifg, v_guibg, v_ctermfg, v_ctermbg],
"   \ [ r_guifg, r_guibg, r_ctermfg, r_ctermbg],
"   \ [ ii_guifg, ii_guibg, ii_ctermfg, ii_ctermbg],
"   \ [ in_guifg, in_guibg, in_ctermfg, in_ctermbg],
" \ ]
" group_a: window id
" group_b/group_c: statusline sections
" group_z: empty area
" group_i: window id in insert mode
" group_v: window id in visual mode
" group_r: window id in select mode
" group_ii: window id in iedit-insert mode
" group_in: windows id in iedit-normal mode
```

OPEN CHAT

```
function! SpaceVim#mapping#guide#theme#gruvbox#palette() abort
    return [
        \ ['#282828', '#a89984', 246, 235],
        \ ['#a89984', '#504945', 239, 246],
        \ ['#a89984', '#3c3836', 237, 246],
        \ ['#665c54', 241],
        \ ['#282828', '#83a598', 235, 109],
        \ ['#282828', '#fe8019', 235, 208],
        \ ['#282828', '#8ec07c', 235, 108],
        \ ['#282828', '#689d6a', 235, 72],
        \ ['#282828', '#8f3f71', 235, 132],
        \ ]
endfunction
```

这一模板是 gruvbox 主题的，当你需要在切换主题时，状态栏都使用同一种颜色主题，可以设置 `custom_color_palette`：

```
[options]
    custom_color_palette = [
        ["#282828", "#a89984", 246, 235],
        ["#a89984", "#504945", 239, 246],
        ["#a89984", "#3c3836", 237, 246],
        ["#665c54", 241],
        ["#282828", "#83a598", 235, 109],
        ["#282828", "#fe8019", 235, 208],
        ["#282828", "#8ec07c", 235, 108],
        ["#282828", "#689d6a", 235, 72],
        ["#282828", "#8f3f71", 235, 132],
    ]
```

标签栏

OPEN CHAT

如果只有一个 Tab, Buffers 将被罗列在标签栏上，每一个包含：序号、文件类型图标、文件名。如果有不止一个 Tab, 那么所有 Tab 将被罗列在标签栏上。标签栏上每一个 Tab 或者 Buffer 可通过快捷键 `<Leader> number` 进行快速访问，默认的 `<Leader>` 是 `\`。

快捷键	功能描述
<code><Leader> 1</code>	跳至标签栏序号 1
<code><Leader> 2</code>	跳至标签栏序号 2
<code><Leader> 3</code>	跳至标签栏序号 3
<code><Leader> 4</code>	跳至标签栏序号 4
<code><Leader> 5</code>	跳至标签栏序号 5
<code><Leader> 6</code>	跳至标签栏序号 6
<code><Leader> 7</code>	跳至标签栏序号 7
<code><Leader> 8</code>	跳至标签栏序号 8
<code><Leader> 9</code>	跳至标签栏序号 9

标签栏上也支持鼠标操作，左键可以快速切换至该标签，中键删除该标签。该特性只支持 Neovim，并且需要 `has('tablineat')` 特性。

快捷键	功能描述
<code><Mouse-left></code>	切换至该标签
<code><Mouse-middle></code>	删除该标签

标签管理器

可使用 `SPC t t` 打开内置的标签管理器，标签管理器内的快捷键如下：

快捷键	功能描述
<code>o</code>	展开或关闭标签目录

OPEN CHAT

快捷键	功能描述
<code>r</code>	重命名光标下的标签页
<code>n</code>	在光标位置下新建命名标签页
<code>N</code>	在光标位置下新建匿名标签页
<code>x</code>	删除光标下的标签页
<code>Ctrl-S-<Up></code>	向上移动光标下的标签页
<code>Ctrl-S-<Down></code>	向下移动光标下的标签页
<code><Enter></code>	跳至光标所对应的标签窗口

基本操作

窗口管理器

窗口管理器快捷键只可以在 Normal 模式下使用，默认的前缀（WIN）按键为 `s`，可以在配置文件中通过修改 SpaceVim 选项 `window_leader` 的值来设为其它按键：

```
[options]
windows_leader = "s"
```

快捷键	功能描述
<code>q</code>	智能关闭当前窗口
<code>WIN v</code>	水平分屏
<code>WIN V</code>	水平分屏，并编辑上一个文件
<code>WIN g</code>	垂直分屏
<code>WIN G</code>	垂直分屏，并编辑上一个文件
<code>WIN t</code>	新建新的标签页

[OPEN CHAT](#)

快捷键	功能描述
<code>WIN o</code>	关闭其他窗口
<code>WIN x</code>	关闭当前缓冲区，并保留新的空白缓冲区
<code>WIN q</code>	关闭当前缓冲区
<code>WIN Q</code>	关闭当前窗口
<code><Tab></code>	跳至下一个窗口
<code>Shift-<Tab></code>	跳至上一个窗口

Normal 模式下的按键 `q` 被用来快速关闭窗口，其原生的功能可以使用 `<Leader> q r` 来代替。

快捷键	模式	功能描述
<code><leader>+y</code>	Visual	Copy selection to X11 clipboard ("y)
<code>Ctrl-c</code>	Normal	Copy full path of current buffer to X11 clipboard
<code><leader>+Ctrl-c</code>	Normal	Copy github.com url of current buffer to X11 clipboard(if it is a github repo)
<code><leader>+Ctrl-l</code>	Normal/Visual	Copy github.com url of current lines to X11 clipboard(if it is a github repo)
<code><leader>+p</code>	Normal/Visual	Paste selection from X11 clipboard ("p)
<code>Ctrl-f</code>	Normal	Smart page forward (C-f/C-d)
<code>Ctrl-b</code>	Normal	Smart page backwards (C-b/C-u)
<code>Ctrl-e</code>	Normal	Smart scroll down (3C-e/j)
<code>Ctrl-y</code>	Normal	Smart scroll up (3C-y/k)
<code>Ctrl-q</code>	Normal	<code>Ctrl-w</code>

OPEN CHAT

快捷键	模式	功能描述
Ctrl-x	Normal	Switch buffer and placement
<Up> / <Down>	Normal	Smart up and down
}	Normal	After paragraph motion go to first non-blank char (}^)
<	Visual/Normal	Indent to left and re-select
>	Visual/Normal	Indent to right and re-select
<Tab>	Visual	Indent to right and re-select
Shift-<Tab>	Visual	Indent to left and re-select
gp	Normal	Select last paste
Q / g Q	Normal	Disable EX-mode ()
Ctrl-a	Command	Navigation in command line
Ctrl-b	Command	Move cursor backward in command line
Ctrl-f	Command	Move cursor forward in command line

文件操作

按键	功能描述
SPC f s / Ctrl-s	保存文件 (:w)
SPC f W	使用管理员模式保存

编辑器界面

按键	功能描述
<F2>	Toggle tagbar

OPEN CHAT

按键	功能描述
<F3>	Toggle Vimfiler
<Leader> + num	Jump to the buffer with the num index
<Alt> + num	Jump to the buffer with the num index, this only works in Neovim
Alt-h / <Left>	Jump to left buffer in the tabline, this only works in Neovim
Alt-l / <Right>	Jump to Right buffer in the tabline, this only works in Neovim
<Leader> t s	Toggle spell-checker (:setlocal spell!)
<Leader> t n	Toggle line numbers (:setlocal nonumber!)
<Leader> t l	Toggle hidden characters (:setlocal nolist!)
<Leader> t h	Toggle highlighted search (:set hlsearch!)
<Leader> t w	Toggle wrap (:setlocal wrap! breakindent!)
g 0	Go to first tab (:tabfirst)
g \$	Go to last tab (:tablast)
g r	Go to previous tab (:tabprevious)
Ctrl-<Down>	Move to split below (Ctrl-w j)
Ctrl-<Up>	Move to upper split (Ctrl-w k)
Ctrl-<Left>	Move to left split (Ctrl-w h)
Ctrl-<Right>	Move to right split (Ctrl-w l)
*	Search selection forwards
#	Search selection backwards
, <Space>	Remove all spaces at EOL

[OPEN CHAT](#)

按键	功能描述
<code>Ctrl-r</code>	Replace selection
<code><Leader> l j</code>	Next on location list
<code><Leader> l k</code>	Previous on location list
<code><Leader> S</code>	Source selection

原生功能

快捷键	功能描述
<code><leader> q r</code>	原生 <code>q</code> 快捷键
<code><leader> q r/</code>	原生 <code>q /</code> 快捷键，打开命令行窗口
<code><leader> q r?</code>	原生 <code>q ?</code> 快捷键，打开命令行窗口
<code><leader> q r:</code>	原生 <code>q :</code> 快捷键，打开命令行窗口

标签管理

在浏览代码时，通常需要给指定位置添加标签，方便快速跳转，在 SpaceVim 中可以使用如下快捷键来管理标签。这一功能需要载入 `tools` 模块：

```
[layers]
  name = "tools"
```

快捷键	功能描述
<code>m a</code>	显示书签列表
<code>m m</code>	切换当前行标签状态
<code>m n</code>	跳至下一个书签
<code>m p</code>	跳至前一个书签

[OPEN CHAT](#)

快捷键	功能描述
<code>m i</code>	给当前行标签添加说明

正因为占用了以上几个快捷键，以下几个寄存器无法用来记忆当前位置了：`a, m, n, p, i`。当然，也可以在启动函数里将 `<Leader> m` 映射为 `m` 键，如此便可使用 `<Leader> m a` 来代替 `m a`。

```
function! myspacevim#before() abort
    nnoremap <silent><Leader>m m
endfunction
```

模糊搜索

目前一共有五种模糊搜索的模块，分别对应不同的工具：

- » denite
- » unite
- » leaderf
- » ctrlp
- » fzf

这些模块都提供了非常类似的快捷键，包括文件搜索、跳转历史搜索等功能，具体快捷键列表如下：

快捷键

快捷键	功能描述
<code><Leader> f <Space></code>	模糊查找快捷键，并执行该快捷键
<code><Leader> f e</code>	模糊搜索寄存器
<code><Leader> f h</code>	模糊搜索 history/yank
<code><Leader> f j</code>	模糊搜索 jump, change
<code><Leader> f l</code>	模糊搜索 location list
<code><Leader> f m</code>	模糊搜索 output messages
<code><Leader> f o</code>	模糊搜索函数列表

OPEN CHAT

快捷键	功能描述
<code><Leader> f q</code>	模糊搜索 quickfix list
<code><Leader> f r</code>	重置上次搜索窗口

但是由于不同工具的局限性，有些模块还不能完全提供上述功能，目前仅有 denite 和 unite 模块可以提供完整的功能。

功能特性	unite	denite	leaderf	ctrlp	fzf
模糊查找快捷键，并执行该快捷键	yes	yes	no	no	no
模块搜索寄存器	yes	yes	no	yes	yes
模糊搜索文件	yes	yes	yes	yes	yes
模糊搜索复制历史	yes	yes	no	no	yes
模糊搜索跳转历史	yes	yes	no	yes	yes
模糊搜索位置列表	yes	yes	no	no	yes
模糊搜索语法树	yes	yes	yes	yes	yes
模糊搜索消息	yes	yes	no	no	yes
模糊搜索全局位置列表	yes	yes	no	yes	yes
重置上次搜索窗口	yes	yes	no	no	no

模糊搜索窗口内的快捷键：

快捷键	功能描述
<code><Tab> / Ctrl-j</code>	下一个选项
<code>Shift-<Tab> / Ctrl-k</code>	上一个选项
<code>jk</code>	离开输入模式（仅支持 denite 和 unite 模块）

[OPEN CHAT](#)

快捷键	功能描述
Ctrl-w	删除光标前词语
<Enter>	执行默认动作
Ctrl-s	在分割窗口内打开
Ctrl-v	在垂直分割窗口内打开
Ctrl-t	在新的标签页里打开
Ctrl-g	推出模糊搜索插件

Denite 或 **Unite** 模块可视模式下快捷键：

快捷键	功能描述
Ctrl+h/k/l/r	未定义
Ctrl+l	刷新窗口
<Tab>	选择即将执行的动作
Space	切换标记当前选项
r	替换或者重命名
Ctrl+z	切换窗口分割方式

以上这些快捷键仅仅是模糊搜索模块的部分快捷键，其他快捷键信息可查阅对应模块文档。

交互

快捷键

快捷键导航

当 Normal 模式下按下前缀键后出现输入延迟，则会在屏幕下方打开一个快捷键导航窗口，提示当前可用的快捷键及其功能描述，目前支持的前缀键有：**[SPC]**、**[Window]**、**<Leader>**、**g**、**z**。

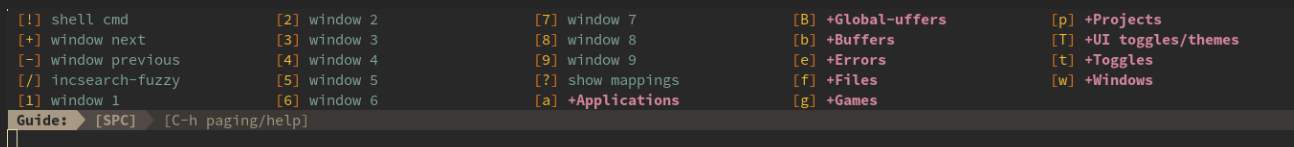
这些前缀的按键为：

OPEN CHAT

前缀名称	用户选项以及默认值	功能描述
[SPC]	空格键	SpaceVim 默认前缀键
[Window]	windows_leader / s	SpaceVim 默认窗口前缀键
<leader>	默认的 Vim leader 键	Vim/Neovim 默认前缀键

默认情况下，快捷键导航将在输入延迟超过 1000ms 后打开，你可以通过修改 Vim 的 'timeoutlen' 选项来修改成适合自己的延迟时间长度。

例如，Normal 模式下按下空格键，你将会看到：



这一导航窗口将提示所有以空格键为前缀的快捷键，并且根据功能将这些快捷键进行了分组，例如 buffer 相关的快捷键都是 b，工程相关的快捷键都是 p。在代码导航窗口内，按下 Ctrl-h 键，可以获取一些帮助信息，这些信息将被显示在状态栏上，提示的是一些翻页和撤销按键的快捷键。

按键	功能描述
u	撤销按键
n	向下翻页
p	向上翻页

如果要自定义以 [SPC] 为前缀的快捷键，可以使用 SpaceVim#custom#SPC()，示例如下：



通过 Unite/Denite 浏览快捷键

可以通过 SPC ? 使用 Unite 将当前快捷键罗列出来。然后可以输入快捷键按键字母或者描述，Unite 可以模糊匹配并展示结果。

OPEN CHAT

```
> format-codo          [SPC]bf
> toggle background    [SPC]tb
> buffer list          [SPC]bb
> Open previous buffer [SPC]bN
> Open previous buffer [SPC]bp
> Open next buffer     [SPC]bn
> alternate-window     [SPC]w<Tab>
[SPC]b
~
~
~
~
~
~
~
Unite default menu:CustomKeyMaps(7/114) | Custom mapped keyboard shortcuts [unite]<SPACE> [unite] - default
-- INSERT --
```

使用 `<Tab>` 键或者上下方向键选择你需要的快捷键，回车将执行这一快捷键。

获取帮助信息

Denite/Unite 是一个强大的信息筛选浏览器，这类似于 Emacs 中的 **Helm**。以下这些快捷键将帮助你快速获取需要的帮助信息：

快捷键	功能描述
<code>SPC h SPC</code>	使用 fuzzy find 模块展示 SpaceVim 帮助文档章节目录
<code>SPC h i</code>	获取光标下单词的帮助信息
<code>SPC h k</code>	使用快捷键导航，展示 SpaceVim 所支持的前缀键
<code>SPC h m</code>	使用 Unite 浏览所有 man 文档

报告一个问题：

快捷键	功能描述
<code>SPC h I</code>	根据模板展示 Issue 所必须的信息

可用模块

所有可用模块可以通过命令 `:SPLayer -l` 或者快捷键 `SPC h l` 来展示。

可用的插件

OPEN CHAT

可通过快捷键 `<leader> l p` 列出所有已安装的插件，支持模糊搜索，回车将使用浏览器打开该插件的官网。

添加用户自定义插件

如果添加来自于 github.com 的插件，可以 `用户名/仓库名` 这一格式，将该插件添加到

`[[custom_plugins]]`，示例如下：

```
[[custom_plugins]]
  name = 'lilydjwg/colorizer'
  merged = false
```

界面元素显示切换

所有的界面元素切换快捷键都以 `[SPC] t` 或 `[SPC] T` 开头，你可以在快捷键导航中查阅所有快捷键。

常规操作

光标移动

光标的移动默认采用 Vi 的默认形式：`hjkl`。

快捷键	功能描述
<code>h</code>	向左移动光标（Vim 原生功能，无映射）
<code>j</code>	向下移动光标（Vim 原生功能，无映射）
<code>k</code>	向上移动光标（Vim 原生功能，无映射）
<code>l</code>	向右移动光标（Vim 原生功能，无映射）
<code>H</code>	光标移至屏幕最上方（Vim 原生功能，无映射）
<code>L</code>	光标移至屏幕最下方（Vim 原生功能，无映射）
<code>SPC j 0</code>	跳至行首（并且标记原始位置）
<code>SPC j \$</code>	跳至行尾（并且标记原始位置）

OPEN CHAT

快捷键	功能描述
<code>SPC t -</code>	锁定光标在屏幕中间 (TODO)

使用 vim-easymotion 快速跳转

快速跳到网址 (TODO)

类似于 Firefox 的 Vimperator 的 `f` 键的功能。

快捷键	功能描述
<code>SPC j u / (o for help buffer)</code>	快速跳到/打开 URL

常用的成对快捷键

快捷键	功能描述
<code>[SPC</code>	在当前行或已选区域上方添加空行
<code>] SPC</code>	在当前行或已选区域下方添加空行
<code>[b</code>	跳至前一 buffer
<code>] b</code>	跳至下一 buffer
<code>[f</code>	跳至文件夹中的前一个文件
<code>] f</code>	跳至文件夹中的下一个文件
<code>[l</code>	跳至前一个错误处
<code>] l</code>	跳至下一个错误处
<code>[c</code>	跳至前一个 vcs hunk (需要 VersionControl 模块)
<code>] c</code>	跳至下一个 vcs hunk (需要 VersionControl 模块)
<code>[q</code>	跳至前一个错误

OPEN CHAT

快捷键	功能描述
<code>] q</code>	跳至下一个错误
<code>[t</code>	跳至前一个标签页
<code>] t</code>	跳至下一个标签页
<code>[w</code>	跳至前一个窗口
<code>] w</code>	跳至下一个窗口
<code>[e</code>	向上移动当前行或者已选择行
<code>] e</code>	向下移动当前行或者已选择行
<code>[p</code>	粘贴至当前行上方
<code>] p</code>	粘贴至当前行下方
<code>g p</code>	选择粘贴的区域

跳转，合并，拆分

以 `SPC j` 为前缀的快捷键主要用作：跳转（jumping），合并（joining），拆分（splitting）。

跳转

快捷键	功能描述
<code>SPC j 0</code>	跳至行首，并且在原始位置留下标签，以便跳回
<code>SPC j \$</code>	跳至行尾，并且在原始位置留下标签，以便跳回
<code>SPC j b</code>	向后回跳
<code>SPC j f</code>	向前跳
<code>SPC j d</code>	跳至当前目录某个文件夹

[OPEN CHAT](#)

快捷键	功能描述
SPC j D	跳至当前目录某个文件夹（在另外窗口展示文件列表）
SPC j i	跳至当前文件的某个函数，使用 Denite 打开语法树
SPC j I	跳至所有 Buffer 的语法树 (TODO)
SPC j j	跳至当前窗口的某个字符 (easymotion)
SPC j J	跳至当前窗口的某两个字符的组合 (easymotion)
SPC j k	跳至下一行，并且对齐下一行
SPC j l	跳至某一行 (easymotion)
SPC j q	show the dumb-jump quick look tooltip (TODO)
SPC j u	跳至窗口某个 URL (TODO)
SPC j v	跳至某个 Vim 函数的定义处 (TODO)
SPC j w	跳至 Buffer 中某个单词 (easymotion)

合并，拆分

快捷键	功能描述
J	合并当前行和下一行
SPC j k	跳至下一行，并且对齐该行
SPC j n	从光标处断开当前行，并且插入空行以及进行对齐
SPC j o	从光标处拆分该行，光标停留在当前行行尾
SPC j s	从光标处拆分 String
SPC j S	从光标处使用换行符拆分 String，并自动缩进新行

OPEN CHAT

窗口操作

窗口操作常用快捷键

每一个窗口都有一个编号，该编号显示在状态栏的最前端，可通过 **SPC 编号** 进行快速窗口跳转。

快捷键	功能描述
SPC 1	跳至窗口 1
SPC 2	跳至窗口 2
SPC 3	跳至窗口 3
SPC 4	跳至窗口 4
SPC 5	跳至窗口 5
SPC 6	跳至窗口 6
SPC 7	跳至窗口 7
SPC 8	跳至窗口 8
SPC 9	跳至窗口 9

窗口操作相关快捷键（以 **SPC w** 为前缀）：

快捷键	功能描述
SPC w <Tab>	在同一标签内进行窗口切换
SPC w =	对齐分离的窗口
SPC w b	force the focus back to the minibuffer (TODO)
SPC w c	进入阅读模式，浏览当前窗口 (需要 tools 模块)
SPC w C	选择某一个窗口，并且进入阅读模式 (需要 tools 模块)
SPC w d	删除一个窗口

OPEN CHAT

快捷键	功能描述
<code>SPC u SPC w d</code>	delete a window and its current buffer (does not delete the file) (TODO)
<code>SPC w D</code>	选择一个窗口并关闭
<code>SPC u SPC w D</code>	delete another window and its current buffer using vim-choosewin (TODO)
<code>SPC w t</code>	toggle window dedication (dedicated window cannot be reused by a mode) (TODO)
<code>SPC w f</code>	toggle follow mode (TODO)
<code>SPC w F</code>	新建一个新的标签页
<code>SPC w h</code>	移至左边窗口
<code>SPC w H</code>	将窗口向左移动
<code>SPC w j</code>	移至下方窗口
<code>SPC w J</code>	将窗口向下移动
<code>SPC w k</code>	移至上方窗口
<code>SPC w K</code>	将窗口向上移动
<code>SPC w l</code>	移至右方窗口
<code>SPC w L</code>	将窗口向右移动
<code>SPC w m</code>	最大化/最小化窗口（最大化相当于关闭其它窗口）(TODO, now only support maximize)
<code>SPC w M</code>	选择窗口进行替换
<code>SPC w o</code>	按序切换标签页
<code>SPC w p m</code>	open messages buffer in a popup window (TODO)

OPEN CHAT

快捷键	功能描述
<code>SPC w p p</code>	close the current sticky popup window (TODO)
<code>SPC w r</code>	顺序切换窗口
<code>SPC w R</code>	逆序切换窗口
<code>SPC w s / SPC w -</code>	水平分割窗口
<code>SPC w S</code>	水平分割窗口，并切换至新窗口
<code>SPC w u</code>	undo window layout (used to effectively undo a closed window) (TODO)
<code>SPC w U</code>	redo window layout (TODO)
<code>SPC w v / SPC w /</code>	垂直分离窗口
<code>SPC w V</code>	垂直分离窗口，并切换至新窗口
<code>SPC w w</code>	切换至前一窗口
<code>SPC w W</code>	选择一个窗口

文件和缓冲区操作

缓冲区操作

缓冲区（Buffer）操作相关快捷键都是以 `SPC b` 为前缀的，以下为常用的缓冲区操作快捷键，主要包括了缓冲区的切换和删除等操作：

快捷键	功能描述
<code>SPC <Tab></code>	切换至前一缓冲区，常用于两个缓冲区来回切换
<code>SPC b .</code>	启用缓冲区临时快捷键
<code>SPC b b</code>	通过模糊搜索工具进行缓冲区切换，需要启用一个模糊搜索工具模块

[OPEN CHAT](#)

快捷键	功能描述
SPC b d	删除当前缓冲区，但保留编辑窗口
SPC u SPC b d	kill the current buffer and window (does not delete the visited file) (TODO)
SPC b D	选择一个窗口，并删除其缓冲区
SPC u SPC b D	kill a visible buffer and its window using ace-window(TODO)
SPC b c	删除其它已保存的缓冲区
SPC b C-d	删除其它所有缓冲区
SPC b C-D	kill buffers using a regular expression(TODO)
SPC b e	清除当前缓冲区内容，需要手动确认
SPC b h	打开欢迎界面, 等同于快捷键 SPC a s
SPC b n	切换至下一个缓冲区，排除特殊插件的缓冲区
SPC b m	打开消息缓冲区
SPC u SPC b m	kill all buffers and windows except the current one(TODO)
SPC b p	切换至前一个缓冲区，排除特殊插件的缓冲区
SPC b P	使用系统剪切板内容替换当前缓冲区
SPC b R	从磁盘重新读取当前缓冲区所对应的文件
SPC b s	switch to the <i>scratch</i> buffer (create it if needed) (TODO)
SPC b w	切换只读权限
SPC b Y	将整个缓冲区复制到系统剪切板

[OPEN CHAT](#)

快捷键	功能描述
<code>z f</code>	Make current function or comments visible in buffer as much as possible (TODO)

新建空白 buffer

快捷键	功能描述
<code>SPC b N h</code>	在左侧新建一个窗口，并在其中新建空白 buffer
<code>SPC b N j</code>	在下方新建一个窗口，并在其中新建空白 buffer
<code>SPC b N k</code>	在上方新建一个窗口，并在其中新建空白 buffer
<code>SPC b N l</code>	在右侧新建一个窗口，并在其中新建空白 buffer
<code>SPC b N n</code>	在当前窗口新建一个空白 buffer

特殊 buffer

在 SpaceVim 中，有很多特殊的 buffer，这些 buffer 是由插件或者 SpaceVim 自身建立的，并不会被列出。

文件操作相关快捷键

文件操作相关的快捷键都是以 `SPC f` 为前缀的：

快捷键	功能描述
<code>SPC f /</code>	使用 <code>find</code> 命令查找文件，支持参数提示
<code>SPC f b</code>	跳至文件书签
<code>SPC f c</code>	copy current file to a different location(TODO)
<code>SPC f C d</code>	修改文件编码 unix -> dos
<code>SPC f C u</code>	修改文件编码 dos -> unix

[OPEN CHAT](#)

快捷键	功能描述
SPC f D	删除文件以及 buffer，需要手动确认
SPC f E	open a file with elevated privileges (sudo edit)(TODO)
SPC f f	打开文件
SPC f F	打开光标下的文件
SPC f o	代开文件树，并定位到当前文件
SPC f R	rename the current file(TODO)
SPC f s	保存文件
SPC f S	保存所有文件
SPC f r	打开文件历史
SPC f t	切换侧栏文件树
SPC f T	打开文件树侧栏
SPC f d	Windows 下显示/隐藏磁盘管理器
SPC f y	复制并显示当前文件的绝对路径

Vim 和 SpaceVim 相关文件

SpaceVim 相关的快捷键均以 **SPC f v** 为前缀，这便于快速访问 SpaceVim 的配置文件：

快捷键	功能描述
SPC f v v	复制并显示当前 SpaceVim 的版本
SPC f v d	打开 SpaceVim 的用户配置文件

文件树

OPEN CHAT

SpaceVim 使用 vimfiler 作为默认的文件树插件，默认的快捷键是 **F3**, SpaceVim 也提供了另外一组快捷键 **SPC f t** 和 **SPC f T** 来打开文件树。如果需要修改默认文件树插件，需要在 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中修改选项 `filemanager`：

```
[options]
# 文件树插件可选值包括：
# - vimfiler (默认)
# - nerdtree
# - defx
filemanager = "defx"
```

SpaceVim 的文件树提供了版本控制信息的接口，但是这一特性需要分析文件夹内容，会使得文件树插件比较慢，因此默认没有打开，如果需要使用这一特性，可向配置文件中加入 `enable_vimfiler_gitstatus = true`，启用后的截图如下：



OPEN CHAT

默认情况下文件树是在窗口的右边打开，如果需要设置文件树默认在左边，需要修改 `filetree_direction` 选项。需要注意的是，当设置文件树在左边时，函数列表 tagbar 将会在右边。

```
[options]
filetree_direction = "left"
```

文件树中的常用操作

文件树中主要以 `hjkl` 为核心，这类似于 `vifm` 中常用的快捷键：

快捷键	功能描述
<code><F3> / SPC f t</code>	切换文件树
文件树内的快捷键	
<code><Left> / h</code>	移至父目录，并关闭文件夹
<code><Down> / j</code>	向下移动光标
<code><Up> / k</code>	向上移动光标
<code><Right> / l</code>	展开目录，或打开文件
<code>N</code>	在光标位置新建文件
<code>y y</code>	复制光标下文件路径至系统剪切板
<code>y Y</code>	复制光标下文件至系统剪切板
<code>P</code>	在光标位置黏贴文件
<code>.</code>	切换显示隐藏文件
<code>s v</code>	分屏编辑该文件
<code>s g</code>	垂直分屏编辑该文件
<code>p</code>	预览文件

OPEN CHAT

快捷键	功能描述
<code>i</code>	切换至文件夹历史
<code>v</code>	快速查看
<code>g x</code>	使用相关程序执行该文件
<code>,</code>	标记光标下的文件（夹）
<code>V</code>	清除所有标记
<code>Ctrl+r</code>	刷新页面

文件树中打开文件

如果只有一个可编辑窗口，则在该窗口中打开选择的文件，否则则需要指定窗口来打开文件：

快捷键	功能描述
<code>l / <Enter></code>	打开文件
<code>sg</code>	分屏打开文件
<code>sv</code>	垂直分屏打开文件

以 `g` 为前缀的快捷键

在 Normal 模式下按下 `g` 之后，如果你不记得快捷键出现按键延迟，那么快捷键导航将会提示你所有以 `g` 为前缀的快捷键。

快捷键	功能描述
<code>g #</code>	反向搜索光标下的词
<code>g \$</code>	跳向本行最右侧字符
<code>g &</code>	针对所有行重复执行上一次 “:s” 命令
<code>g '</code>	跳至标签

OPEN CHAT

快捷键	功能描述
<code>g *</code>	正向搜索光标下的词
<code>g +</code>	newer text state
<code>g ,</code>	newer position in change list
<code>g -</code>	older text state
<code>g /</code>	stay incsearch
<code>g 0</code>	go to leftmost character
<code>g ;</code>	older position in change list
<code>g <</code>	last page of previous command output
<code>g <Home></code>	go to leftmost character
<code>g E</code>	end of previous word
<code>g F</code>	edit file under cursor(jump to line after name)
<code>g H</code>	select line mode
<code>g I</code>	insert text in column 1
<code>g J</code>	join lines without space
<code>g N</code>	visually select previous match
<code>g Q</code>	switch to Ex mode
<code>g R</code>	enter VREPLACE mode
<code>g T</code>	previous tag page
<code>g U</code>	make motion text upercase
<code>g]</code>	tselect cursor tag

[OPEN CHAT](#)

快捷键	功能描述
<code>g ^</code>	go to leftmost no-white character
<code>g _</code>	go to last char
<code>g `</code>	跳至标签，等同于 <code>g '</code>
<code>g a</code>	打印光标字符的 ascii 值
<code>g d</code>	跳至定义处
<code>g e</code>	go to end of previous word
<code>g f</code>	edit file under cursor
<code>g g</code>	go to line N
<code>g h</code>	select mode
<code>g i</code>	insert text after '^ mark
<code>g j</code>	move cursor down screen line
<code>g k</code>	move cursor up screen line
<code>g m</code>	go to middle of screenline
<code>g n</code>	visually select next match
<code>g o</code>	goto byte N in the buffer
<code>g s</code>	sleep N seconds
<code>g t</code>	next tag page
<code>g u</code>	make motion text lowercase
<code>g ~</code>	swap case for Nmove text
<code>g <End></code>	跳至本行最右侧字符，等同于 <code>g \$</code>

[OPEN CHAT](#)

快捷键	功能描述
<code>g Ctrl-G</code>	显示光标信息

以 `z` 开头的命令

当你不记得按键映射时，你可以在普通模式下输入前缀 `z`，然后你会看到所有以 `z` 为前缀的函数映射。

快捷键	功能描述
<code>z <Right></code>	scroll screen N characters to left
<code>z +</code>	cursor to screen top line N
<code>z -</code>	cursor to screen bottom line N
<code>z .</code>	cursor line to center
<code>z <Cr></code>	cursor line to top
<code>z =</code>	spelling suggestions
<code>z A</code>	toggle folds recursively
<code>z C</code>	close folds recursively
<code>z D</code>	delete folds recursively
<code>z E</code>	eliminate all folds
<code>z F</code>	create a fold for N lines
<code>z G</code>	mark good spelled(update internal-wordlist)
<code>z H</code>	scroll half a screenwidth to the right
<code>z L</code>	scroll half a screenwidth to the left
<code>z M</code>	set <code>foldlevel</code> to zero
<code>z N</code>	set <code>foldenable</code>

OPEN CHAT

快捷键	功能描述
z O	open folds recursively
z R	set <code>foldlevel</code> to deepest fold
z W	mark wrong spelled
z X	re-apply <code>foldlevel</code>
z ^	cursor to screen bottom line N
z a	toggle a fold
z b	redraw, cursor line at bottom
z c	close a fold
z d	delete a fold
z e	right scroll horizontally to cursor position
z f	create a fold for motion
z g	mark good spelled
z h	scroll screen N characters to right
z i	toggle foldenable
z j	mode to start of next fold
z k	mode to end of previous fold
z l	scroll screen N characters to left
z m	subtract one from <code>foldlevel</code>
z n	reset <code>foldenable</code>
z o	open fold

OPEN CHAT

快捷键	功能描述
<code>z r</code>	add one to <code>foldlevel</code>
<code>z s</code>	left scroll horizontally to cursor position
<code>z t</code>	cursor line at top of window
<code>z v</code>	open enough folds to view cursor line
<code>z x</code>	re-apply foldlevel and do "zV"
<code>z z</code>	smart scroll
<code>z <Left></code>	scroll screen N characters to right

搜索

使用额外工具

SpaceVim 像下面那样调用不同搜索工具的搜索接口：

- » `rg - ripgrep`
- » `ag - the silver searcher`
- » `pt - the platinum searcher`
- » `ack`
- » `grep`

SpaceVim 中的搜索命令以 `SPC s` 为前缀，前一个键是使用的工具，后一个键是范围。例如 `SPC s a b` 将使用 `ag` 在当前所有已经打开的缓冲区中进行搜索。

如果最后一个键（决定范围）是大写字母，那么就会对当前光标下的单词进行搜索。举个例子 `SPC s a B` 将会搜索当前光标下的单词。

如果工具键被省略了，那么会用默认的搜索工具进行搜索。默认的搜索工具对应在 `search_tools` 列表中的第一个工具。列表中的工具默认的顺序为：`rg, ag, pt, ack, grep`。举个例子：如果 `rg` 和 `ag` 没有在系统中找到，那么 `SPC s b` 会使用 `pt` 进行搜索。

下表是全部的工具键：

[OPEN CHAT](#)

工具	键
ag	a
grep	g
ack	k
rg	r
pt	t

应当避免的范围和对应按键为：

范围	键
打开的缓冲区	b
给定目录的文件	f
当前工程	p

可以双击按键序列中的第二个键来在当前文件中进行搜索。举个例子：SPC s a a 会使用 **ag** 在当前文件中进行搜索。

注意：

- » 如果使用源代码管理的话 **rg**, **ag** 和 **pt** 都会被忽略掉，但是他们可以在任意目录中正常运行。
- » 也可以通过将它们标记在联合缓冲区来一次搜索多个目录。注意 如果你使用 **pt**, **TCL parser tools** 同时也需要安装一个名叫 **pt** 的命令行工具。

配置搜索工具

若需要修改默认搜索工具的选项，可以使用启动函数，在启动函数中配置各种搜索工具的默认选项。下面是一个修改 **rg** 默认搜索选项的配置示例：

```
function! myspacevim#before() abort
    let profile = SpaceVim#mapping#search#getprofile('rg')
    let default_opt = profile.default_opts + ['--no-ignore-vcs']
```

OPEN CHAT


```
call SpaceVim#mapping#search#profile({'rg' : {'default_opts' : default_opt}})
endfunction
```

搜索工具配置结构为：

```
" { 'ag' : {
"   'namespace' : '',      " a single char a-z
"   'command' : '',        " executable
"   'default_opts' : [],   " default options
"   'recursive_opt' : [],  " default recursive options
"   'expr_opt' : '',       " option for enable expr mode
"   'fixed_string_opt' : '', " option for enable fixed string mode
"   'ignore_case' : '',    " option for enable ignore case mode
"   'smart_case' : '',     " option for enable smart case mode
" }
" }
```

常用按键绑定

快捷键	功能描述
<code>SPC r l</code>	resume the last completion buffer
<code>SPC s `</code>	go back to the previous place before jump
Prefix argument	will ask for file extensions

在当前文件中进行搜索

快捷键	功能描述
<code>SPC s s</code>	search with the first found tool
<code>SPC s S</code>	search with the first found tool with default input

OPEN CHAT

快捷键	功能描述
SPC s a a	ag
SPC s a A	ag with default input
SPC s g g	grep
SPC s g G	grep with default input
SPC s r r	rg
SPC s r R	rg with default input

搜索当前文件所在的文件夹

快捷键	功能描述
SPC s d	searching in buffer directory with default tool
SPC s D	searching in buffer directory cursor word with default tool
SPC s a d	searching in buffer directory with ag
SPC s a D	searching in buffer directory cursor word with ag
SPC s g d	searching in buffer directory with grep
SPC s g D	searching in buffer directory cursor word with grep
SPC s k d	searching in buffer directory with ack
SPC s k D	searching in buffer directory cursor word with ack
SPC s r d	searching in buffer directory with rg
SPC s r D	searching in buffer directory cursor word with rg
SPC s t d	searching in buffer directory with pt
SPC s t D	searching in buffer directory cursor word with pt

OPEN CHAT

在所有打开的缓冲区中进行搜索

快捷键	功能描述
SPC s b	search with the first found tool
SPC s B	search with the first found tool with default input
SPC s a b	ag
SPC s a B	ag with default input
SPC s g b	grep
SPC s g B	grep with default input
SPC s k b	ack
SPC s k B	ack with default input
SPC s r b	rg
SPC s r B	rg with default input
SPC s t b	pt
SPC s t B	pt with default input

在任意目录中进行搜索

快捷键	功能描述
SPC s f	search with the first found tool
SPC s F	search with the first found tool with default input
SPC s a f	ag
SPC s a F	ag with default text

OPEN CHAT

快捷键	功能描述
SPC s g f	grep
SPC s g F	grep with default text
SPC s k f	ack
SPC s k F	ack with default text
SPC s r f	rg
SPC s r F	rg with default text
SPC s t f	pt
SPC s t F	pt with default text

在工程中进行搜索

快捷键	功能描述
SPC / / SPC s p	search with the first found tool
SPC * / SPC s P	search with the first found tool with default input
SPC s a p	ag
SPC s a P	ag with default text
SPC s g p	grep
SPC s g P	grep with default text
SPC s k p	ack
SPC s k P	ack with default text
SPC s t p	pt
SPC s t P	pt with default text

OPEN CHAT

快捷键	功能描述
SPC s r p	rg
SPC s r P	rg with default text

提示: 在工程中进行搜索的话, 无需提前打开文件。在工程保存目录中使用 SPC p p 和 C-s, 就比如 SPC s p。 (TODO)

后台进行工程搜索

在工程中进行后台搜索时, 当搜索完成时, 会在状态栏上进行显示。

快捷键	功能描述
SPC s j	searching input expr background with the first found tool
SPC s J	searching cursor word background with the first found tool
SPC s l	List all searching result in quickfix buffer
SPC s a j	ag
SPC s a J	ag with default text
SPC s g j	grep
SPC s g J	grep with default text
SPC s k j	ack
SPC s k J	ack with default text
SPC s t j	pt
SPC s t J	pt with default text
SPC s r j	rg
SPC s r J	rg with default text

OPEN CHAT

在网上进行搜索

快捷键	功能描述
<code>SPC s w g</code>	Get Google suggestions in Vim. Opens Google results in Browser.
<code>SPC s w w</code>	Get Wikipedia suggestions in Vim. Opens Wikipedia page in Browser.(TODO)

注意: 为了在 Vim 中使用谷歌 suggestions, 需要在 `~/.SpaceVim.d/init.toml` 的 `[options]` 片段中加入如下配置:

```
[options]
enable_googlesuggest = true
```

实时代码检索

快捷键	功能描述
<code>SPC s g G</code>	在工程中使用默认工具实时检索代码

FlyGrep 缓冲区的按键绑定:

快捷键	功能描述
<code><Esc></code>	close FlyGrep buffer
<code><Enter></code>	open file at the cursor line
<code><Tab></code>	move cursor line down
<code>Shift-<Tab></code>	move cursor line up
<code><Backspace></code>	remove last character
<code>Ctrl-w</code>	remove the Word before the cursor
<code>Ctrl-u</code>	remove the Line before the cursor
<code>Ctrl-k</code>	remove the Line after the cursor

[OPEN CHAT](#)

快捷键	功能描述
<code>Ctrl-a / <Home></code>	Go to the beginning of the line
<code>Ctrl-e / <End></code>	Go to the end of the line

保持高亮

SpaceVim 使用 `search_highlight_persist` 保持当前搜索结果的高亮状态到下一次搜索。同样可以通过 `SPC s c` 或者运行 命令 `:nohlsearch` 来取消搜索结果的高亮表示。

高亮光标下变量

SpaceVim supports highlighting of the current symbol on demand and add a transient state to easily navigate and rename these symbol.

It is also possible to change the range of the navigation on the fly to:

- » buffer
- » function
- » visible area

使用快捷键 `SPC s h` 来高亮光标下的符号。

Navigation between the highlighted symbols can be done with the commands:

快捷键	功能描述
<code>*</code>	initiate navigation transient state on current symbol and jump forwards
<code>#</code>	initiate navigation transient state on current symbol and jump backwards
<code>SPC s e</code>	edit all occurrences of the current symbol
<code>SPC s h</code>	highlight the current symbol and all its occurrence within the current range
<code>SPC s H</code>	go to the last searched occurrence of the last highlighted symbol

In highlight symbol transient state:

快捷键	功能描述
-----	------

OPEN CHAT

快捷键	功能描述
e	edit occurrences (*)
n	go to next occurrence
N / p	go to previous occurrence
b	search occurrence in all buffers
/	search occurrence in whole project
<Tab>	toggle highlight current occurrence
r	change range (function, display area, whole buffer)
R	go to home occurrence (reset position to starting occurrence)
Any other key	leave the navigation transient state

编辑

粘贴文本

粘贴文本自动缩进

文本操作命令

文本相关的命令 (以 x 开头) :

快捷键	功能描述
SPC x a &	基于分隔符 & 进行文本对齐
SPC x a (基于分隔符 (进行文本对齐
SPC x a)	基于分隔符) 进行文本对齐
SPC x a [基于分隔符 [进行文本对齐

OPEN CHAT

快捷键	功能描述
SPC x a]	基于分隔符] 进行文本对齐
SPC x a {	基于分隔符 { 进行文本对齐
SPC x a }	基于分隔符 } 进行文本对齐
SPC x a ,	基于分隔符 , 进行文本对齐
SPC x a .	基于分隔符 . 进行文本对齐(for numeric tables)
SPC x a :	基于分隔符 : 进行文本对齐
SPC x a ;	基于分隔符 ; 进行文本对齐
SPC x a =	基于分隔符 = 进行文本对齐
SPC x a	基于分隔符 进行文本对齐
SPC x a	基于分隔符 进行文本对齐
SPC x a SPC	基于分隔符 进行文本对齐
SPC x a a	align region (or guessed section) using default rules (TODO)
SPC x a c	align current indentation region using default rules (TODO)
SPC x a l	left-align with evil-lion (TODO)
SPC x a L	right-align with evil-lion (TODO)
SPC x a r	基于用户自定义正则表达式进行文本对齐
SPC x a o	对齐算术运算符 +-*/
SPC x c	统计选中区域的字符/单词/行数
SPC x d w	删除行尾空白字符
SPC x d SPC	Delete all spaces and tabs around point, leaving one space

[OPEN CHAT](#)

快捷键	功能描述
SPC x g l	set lanuages used by translate commands (TODO)
SPC x g t	使用 Google Translate 翻译当前单词
SPC x g T	reverse source and target languages (TODO)
SPC x i c	change symbol style to <code>lowerCamelCase</code>
SPC x i C	change symbol style to <code>UpperCamelCase</code>
SPC x i i	cycle symbol naming styles (i to keep cycling)
SPC x i -	change symbol style to <code>kebab-case</code>
SPC x i k	change symbol style to <code>kebab-case</code>
SPC x i _	change symbol style to <code>under_score</code>
SPC x i u	change symbol style to <code>under_score</code>
SPC x i U	change symbol style to <code>UP_CASE</code>
SPC x j c	居中对齐当前段落
SPC x j f	set the justification to full (TODO)
SPC x j l	左对齐当前段落
SPC x j n	set the justification to none (TODO)
SPC x j r	右对齐当前段落
SPC x J	将当前行向下移动一行并进入临时快捷键状态
SPC x K	将当前行向上移动一行并进入临时快捷键状态
SPC x l d	duplicate line or region (TODO)
SPC x l s	sort lines (TODO)

[OPEN CHAT](#)

快捷键	功能描述
SPC x l u	uniquify lines (TODO)
SPC x o	use avy to select a link in the frame and open it (TODO)
SPC x O	use avy to select multiple links in the frame and open them (TODO)
SPC x t c	交换当前字符和前一个字符的位置
SPC x t C	交换当前字符和后一个字符的位置
SPC x t w	交换当前单词和前一个单词的位置
SPC x t W	交换当前单词和后一个单词的位置
SPC x t l	交换当前行和前一行的位置
SPC x t L	交换当前行和后一行的位置
SPC x u	将选中字符串转为小写
SPC x U	将选中字符串转为大写
SPC x w c	统计选中区域的单词数
SPC x w d	show dictionary entry of word from wordnik.com (TODO)
SPC x <Tab>	indent or dedent a region rigidly (TODO)

文本插入命令

文本插入相关命令（以 i 开头）：

快捷键	功能描述
SPC i l l	insert lorem-ipsum list
SPC i l p	insert lorem-ipsum paragraph
SPC i l s	insert lorem-ipsum sentence

OPEN CHAT

快捷键	功能描述
SPC i p 1	insert simple password
SPC i p 2	insert stronger password
SPC i p 3	insert password for paranoids
SPC i p p	insert a phonetically easy password
SPC i p n	insert a numerical password
SPC i u	Search for Unicode characters and insert them into the active buffer.
SPC i U 1	insert UUIDv1 (use universal argument to insert with CID format)
SPC i U 4	insert UUIDv4 (use universal argument to insert with CID format)
SPC i U U	insert UUIDv4 (use universal argument to insert with CID format)

增加或减小数字

快捷键	功能描述
SPC n +	为光标下的数字加 1 并进入 临时快捷键状态
SPC n -	为光标下的数字减 1 并进入 临时快捷键状态

在临时快捷键模式下：

快捷键	功能描述
+	为光标下的数字加 1
-	为光标下的数字减 1
其它任意键	离开临时快捷键状态

提示：如果你想为光标下的数字所增加的值大于 1，你可以使用前缀参数。例如：**10** SPC n + 将为光标下的数字加 **10**。

OPEN CHAT

ledit 多光标编辑

SpaceVim 内置了 leedit 多光标模式，可快速进行多光标编辑。这一功能引入了两个新的模式：`iedit-Normal` 模式和 `iedit-Insert`。

`iedit` 模式默认的颜色是 `red/green`，但它也基于当前的主题。

ledit 快捷键

模式转换:

前面提到 leedit 引入了两个新的模式，在这两个新的模式以及 Vim 自身模式之间转换的快捷键如下：

快捷键	From	to
<code>SPC s e</code>	Normal/Visual	<code>iedit-Normal</code>
<code><Esc></code>	<code>iedit-Normal</code>	Normal

在 `iedit-Normal` 模式中：

`iedit-Normal` 模式继承自 Vim 的 Normal 模式, 下面所列举的是 `iedit-Normal` 模式专属的快捷键。

快捷键	功能描述
<code><Esc></code>	切换回 Normal 模式
<code>i</code>	切换至 <code>iedit-Insert</code> 模式，类似于一般模式下的 <code>i</code>
<code>a</code>	切换至 <code>iedit-Insert</code> 模式，类似于一般模式下的 <code>a</code>
<code>I</code>	跳至当前 occurrence 并进入 <code>iedit-Insert</code> 模式，类似于一般模式下的 <code>I</code>
<code>A</code>	跳至当前 occurrence 并进入 <code>iedit-Insert</code> 模式，类似于一般模式下的 <code>A</code>
<code><Left> / h</code>	左移光标，类似于一般模式下的 <code>h</code>
<code><Right> / l</code>	右移光标，类似于一般模式下的 <code>l</code>
<code>0 / <Home></code>	跳至当前 occurrence 的开头，类似于一般模式下的 <code>0</code>

OPEN CHAT

快捷键	功能描述
<code>\$ / <End></code>	跳至当前 occurrence 的结尾，类似于一般模式下的 <code>\$</code>
<code>C</code>	删除所有 occurrences 中从光标位置开始到 occurrences 结尾的字符，类似于一般模式下的 <code>C</code>
<code>D</code>	删除所有 occurrences 类似于一般模式下的 <code>D</code>
<code>s</code>	删除所有 occurrences 中光标下的字符并进入 <code>iedit-Insert</code> 模式，类似于一般模式下的 <code>s</code>
<code>S</code>	删除所有 occurrences 并进入 <code>iedit-Insert</code> 模式，类似于一般模式下的 <code>S</code>
<code>x</code>	删除所有 occurrences 中光标下的字符，类似于一般模式下的 <code>x</code>
<code>X</code>	删除所有 occurrences 中光标前的字符，类似于一般模式下的 <code>X</code>
<code>gg</code>	跳至第一个 occurrence，类似于一般模式下的 <code>gg</code>
<code>G</code>	跳至最后一个 occurrence，类似于一般模式下的 <code>G</code>
<code>n</code>	跳至下一个 occurrence
<code>N</code>	跳至上一个 occurrence
<code>p</code>	替换所有 occurrences 为最后复制的文本
<code><Tab></code>	toggle current occurrence

In iedit-Insert mode:

快捷键	功能描述
<code>Ctrl-g / <Esc></code>	回到 <code>iedit-Normal</code> 模式
<code>Ctrl-b / <Left></code>	左移光标
<code>Ctrl-f / <Right></code>	右移光标

OPEN CHAT

快捷键	功能描述
Ctrl-a / <Home>	跳至当前 occurrence 的开头
Ctrl-e / <End>	跳至当前 occurrence 的结尾
Ctrl-w	删除光标前的词
Ctrl-k	删除光标后的词
Ctrl-u	删除光标前所有字符
Ctrl-h / <BackSpace>	删除光标前字符
<Delete>	删除光标后字符

注释 (Commentings)

注释 (comment) 通过工具 [nerdcommenter](#) 来处理，它用下面的按键来界定范围。

快捷键	功能描述
SPC ;	进入注释操作模式
SPC c h	隐藏/显示注释
SPC c l	注释/反注释当前行
SPC c L	注释行
SPC c u	反注释行
SPC c p	注释/反注释段落
SPC c P	注释段落
SPC c s	使用完美格式注释
SPC c t	注释/反注释到行
SPC c T	注释到行

OPEN CHAT

快捷键	功能描述
<code>SPC c y</code>	toggle comment and yank(TODO)
<code>SPC c Y</code>	复制到未命名寄存器后注释
<code>SPC c \$</code>	从光标位置开始注释当前行

小提示：

用 `SPC ;` 可以启动一个 comment operator 模式，在该模式下，可以使用移动命令确认注释的范围，比如 `SPC ; 4 j`，这个组合键会注释当前行以及下方的 4 行。这个数字即为相对行号，可在左侧看到。

多方式编码

SpaceVim 默认使用 `utf-8` 码进行编码。下面是 `utf-8` 编码的四个设置：

- » `fileencodings (fencs) : ucs-bom, utf-8, default, latin1`
- » `fileencoding (fenc) : utf-8`
- » `encoding (enc) : utf-8`
- » `termencoding (tenc) : utf-8 (only supported in Vim)`

修复混乱的显示：`SPC e a` 是自动选择文件编码的按键映射。在选择好文件编码方式后，你可以运行下面的代码来修复编码：

```
set enc=utf-8
write
```

异步运行器和交互式编程

SpaceVim 提供了一个异步执行命令和交互式编程的插件，在大多数语言模块中，已经为该语言定义了默认的执行命令，通常快捷键为 `SPC l r`。如果需要添加额外的命令，可以使用启动函数。比如：添加使用 `F5` 按键异步编译当前项目。

```
nnoremap <silent> <F5> :call SpaceVim#plugins#runner#open('make')
```

目前，SpaceVim 支持如下特性：

- » 使用默认命令一键运行当前文件

OPEN CHAT

- » 使用系统文件管理器选择文件并执行
- » 根据文件顶部标识，选择合适解析器
- » 中断代码运行
- » 底部窗口异步展示运行结果
- » 设置默认的运行语言
- » 选择指定语言来运行
- » 支持交互式编程
- » 运行选择的代码片段

错误处理

SpaceVim 通过默认通过 **checkers** 模块来进行文件语法检查，默认在保存文件时进行错误检查。

错误管理导航键 (以 **e** 开头)：

快捷键	功能描述
SPC t s	切换语法检查器
SPC e c	清除所有错误
SPC e h	describe a syntax checker
SPC e l	切换显示错误/警告列表
SPC e n	跳至下一错误
SPC e p	跳至上一个错误
SPC e v	verify syntax checker setup (useful to debug 3rd party tools configuration)
SPC e .	错误暂态 (error transient state)

下一个/上一个错误导航键和错误暂态(error transient state) 可用于浏览语法检查器和位置列表缓冲区的错误，甚至可检查 Vim 位置列表的所有错误。这包括下面的例子：在已被保存的位置列表缓冲区进行搜索。默认提示符：

提示符	描述	自定义选项
-----	----	-------

OPEN CHAT

提示符	描述	自定义选项
✖	Error	error_symbol
➤	warning	warning_symbol
ⓘ	Info	info_symbol

quickfix 列表移动：

快捷键	功能描述
<Leader> q l	打开 quickfix 列表窗口
<Leader> q c	清除 quickfix 列表
<Leader> q n	跳到 quickfix 列表中下一个位置
<Leader> q p	跳到 quickfix 列表中上一个位置

工程管理

SpaceVim 中的工程通过 vim-projectionist 和 vim-rooter 进行管理。当发现一个 `.git` 目录或 在文件树中发现 `.projections.json` 文件后 vim-rooter 会自动找到项目的根目录。

工程管理的命令以 `p` 开头：

快捷键	功能描述
SPC p '	在当前工程的根目录打开 shell（需要 shell 模块）

在工程中搜索文件

快捷键	功能描述
SPC p f	在当前工程中查找文件
SPC p /	在当前工程中搜索文本内容
SPC p k	关闭当前工程的所有缓冲区

OPEN CHAT

快捷键	功能描述
SPC p t	自动查找工程根目录
SPC p p	显示所有工程

格式规范

SpaceVim 添加了 **EditorConfig** 支持，通过一个配置文件来为不同的文件格式设置对应的代码格式规范，这一工具兼容多种文本编辑器和集成开发环境。

更多配置方式，可以阅读其官方文档：[editorconfig-vim package's documentation](#)。

Vim 服务

SpaceVim 在启动时启动了一个服务器。无论何时，当你关闭了 Vim 窗口，该服务器就会被关闭。

连接到 **Vim** 服务器

如果你使用 Neovim, 你需要安装 **neovim-remote**，然后增加如下配置到你的 `bashrc`。

```
export PATH=$PATH:$HOME/.SpaceVim/bin
```

使用命令 **svc** 在一个已存在的 Vim 服务器上打开文件，使用命令 **nsvc** 在一个已存在的 Neovim 服务器上打开文件。

OPEN CHAT

```
1 index.md 2 java.md Buffers
1 ---
2 title: "SpaceVim lang#java layer"
3 ---
4 # [SpaceVim Layers:](https://spacevim.org/layers) lang#java
5
6 <!-- vim-markdown-toc GFM -->
7 * [Description](#description)
8 * [Layer Installation](#layer-installation)
9 * [Key bindings](#key-bindings)
10 * [Java language specified key bindings](#java-language-specified-
    key-bindings)
11 * [Maven](#maven)
12 * [Jump](#jump)
13 * [Problems buffer](#problems-buffer)
14 * [Project buffer](#project-buffer)
15
16 k java.md markdown 1:2 Top
17
18 wsdjeg@archlinux:~$ # vim one the top left, neovim on the top right

1 main.vim Buffers
1 let g:Config_Main_Home = fnamemodify(expand('<sfile>'),
2 \ ':p:h:gs?\|?'.((has('win16') || has('win32')
3 \ || has('win64'))?'\'/' . '?')
4
5
6 " [dir?, path]
7 function! s:parser_argv() abort
8 if !argc()
9 return [1, getcwd()]
10 elseif argv(0) =~# '/'
11 let f = expand(argv(0))
12 if isdirectory(f)
13 return [1, f]
14 else
15 return [1, getcwd()]
16 endif
17
```

Achievements

issues

Achievements	Account
100th issue(issue)	BenBergman
1000th issue(PR)	sei40kr
2000th issue(PR)	nikolaussucher

Stars, forks and watchers

Achievements	Account
First stargazers	monkeydterry

OPEN CHAT

Achievements	Account
100th stargazers	robertofarrell
1000th stargazers	linsongze
2000th stargazers	fated
3000th stargazers	urso
4000th stargazers	wanghe4096
5000th stargazers	xxxxha
6000th stargazers	corenel
7000th stargazers	mohab1989
8000th stargazers	chocopowwwa
9000th stargazers	mffathurr

网站源码位于 Github, [帮助改进本页面](#) — 网站主题: [mattgraham](#),

OPEN CHAT