编辑器vim

vi编辑器

- 早期Unix采用行编辑器ed，后来更新为ex
- 1976年Bill Joy创建面向屏幕的vi编辑器(visual editor)
- 1988年荷兰程序员Bram Moolenaar创建Vim(Vi improved)，向后兼容vi(:set compatible)
- ubuntu发行版预装的vi是最小版的vim-tiny，建议安装完整版的vim或者图形版本gvim

sudo apt install vim　　sudo apt install vim-gtk3

- vi --version 可以查看vi的版本、所支持的特性以及相应的vim配置文件所在位置

✓ user vimrc file: "$HOME/.vimrc"

✓ 2nd user vimrc file: "~/.vim/vimrc"

- vimtutor命令给出了一个简单的教程，学习如何使用vim
- vi [arguments] [file ..] 编辑文件时，正在编辑的文件加载到编辑缓冲区(editing buffer)

✓ 为了防止丢失，会定期将缓冲区内容保存到临时文件(交换文件 .file.swp)

✓ 保存时将当前缓冲区的内容同步到外存

$ cp /etc/hosts .
$ vi hosts

```
127.0.0.1       localhost
127.0.1.1       mars
202.120.224.26  dns

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"hosts" 10L, 239C                                             1,1           All
```

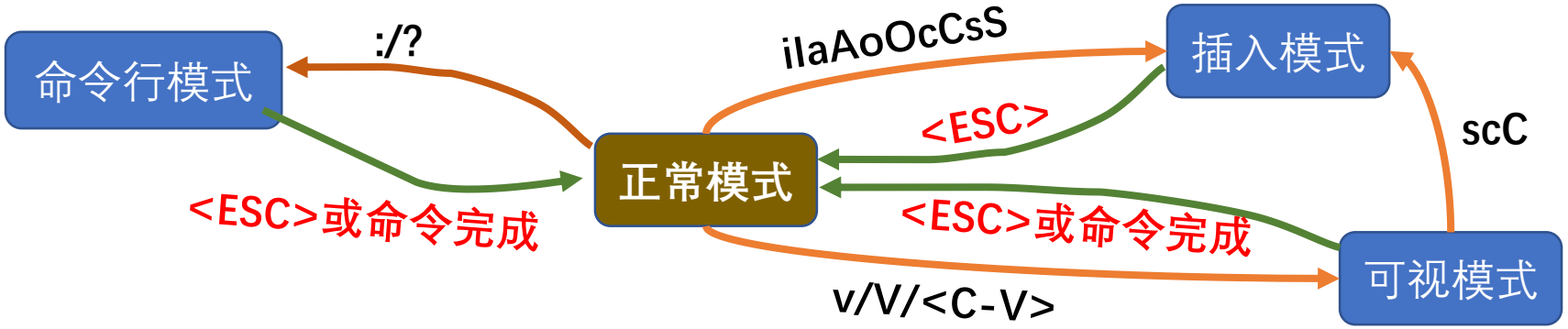
文件内容结束但不够一屏时用~行指示，不属于文件

:set laststatus=2
总是显示状态栏

:set ruler 状态栏包含行和列等

vi编辑器: 模式

- vi是一种模态编辑器，总共有7个模式，比较常用的有4个模式
  - **正常(normal)或命令(command)模式：**vi进入所处的模式，可以执行浏览（光标移动）、编辑（删除和修改和插入文本等）等命令
  - **插入(insert)模式**：键入的内容会插入到缓冲区
  - **可视(visual mode)模式**: 与正常模式相似，通过高亮选择一块文本后进行操作
  - **命令行(command-line/cmdline/last line)模式**: 在窗口底部输入一行命令
    - Ex命令(:)，查找(/ *)和调用外部命令(!)
    - 可以操作文件，改变编辑器本身状态等
  - 所有模式中，正常模式是中心，可以通过**<ESC>或<C-[>**从其他模式切换到正常模式



vi选项

- :mk[exrc] [file] 保存当前设置到当前目录下的.exrc或者指定文件
- :mkv[imrc] [file] 和mk类似，只是保存到.vimrc

- vi支持许多选项，允许用户控制vi操作的各种行为
- :set all查看所有的选项; :set查看修改了默认值的选项；:set option?查看选项option的值; set option& 重设为缺省值
- 选项有两种类型:
  - 开关(布尔型)选项，通过:set [no]option设置，通过set option! 开关(toggle)选项
  - 变量(整数和字符串)选项, 通过:set option=value :set opt+=v :set opt-=v 设置
  - 在一行可以设置多个变量，之间以空格隔开就可以了 :set number ruler
- :options 打开选项窗口，分类显示了当前选项的设置和说明，允许调整选项
- :h[elp] options 查看选项帮助，:help 'option' 查看某个选项的帮助
- 选项名字有完整名和简写名, 比如laststatus和ls，两者都可以使用
- 命令也是如此，比如:help 和:h
- 窗口可通过 :close或<C-W>c关闭，可通过<C-W>w或<C-W><C-W>在窗口间切换

number,nu	显示行号	showmode,smd	状态栏下方显示当前模式
ruler,ru	右下方状态栏中显示光标所在位置	cursorline,cul	当前行高亮显示
showcmd,sc	状态栏下方显示当前键入指令	nocompatible, nocp	关闭兼容模式
laststatus,ls	可取0-2，分别表示没有状态栏，多窗口时有，总是有状态栏(set statusline=...定制)		

文件类型自动检测和语法高亮

- vim可以自动检测文件类型，从而为该语言的文件提供支持， :help filetype
  - :filetype on 启动文件类型检测，基于文件后缀以及文件内容
  - :set filetype=type 手动设置文件类型
  - 针对每种文件类型，vim本身提供了一些用vim脚本语言(VimL)编写的内置插件，使得vim的一些命令的行为进行调整以适配该语言，方便编辑
  - :filetype plugin on 启动fieltype插件支持 或 :filetype plugin off 关闭filetype插件支持
  - 不同语言的缩进方式也不一样，vim针对每个语言也提供了插件缩进支持
  - :filetype plugin indent on　:filetype plugin indent off
- 语法高亮 :h[elp] syntax
  - :syntax enable 允许语法高亮 :syntax off 关闭语法高亮
  - :syntax on 启动语法高亮，设置为缺省值
  - :highlight 查看当前的语法高亮设置，也可以进行修改

命令行历史记录
- :history 查看
- 上下箭头以及<C-N> <C-P>浏览

- colo[rscheme] 修改highlight设置。 set background=dark或light，根据窗口背景色调整该选项
  - :colorshceme 查看当前的colorscheme； :colorscheme murphy 设置colorscheme
- 命令行模式支持历史记录，也支持自动完成，Ctrl-D列出/Tab自动完成选择匹配项
  - :set wildmenu　通过菜单形式显示可选项，缺省wildmode设置为full
  - :set wildmode=list:longest,full 控制自动完成行为, <Tab>列出， 第2次<Tab>wildmenu

vi帮助

- :h[elp] 查看帮助
- :help xxx 查看xxx对应的帮助信息, help index查看所有命令

:help <….>	描述	示例
subject	正常模式命令	help i
i_...	插入模式命令	help i_CTRL-W
v_...	可视模式命令	help v_o
c_...	命令行编辑	help c_%
:…	ex命令	help :s
g_	操作符待决模式	help g_CTRL-A
/	正则表达式	help /\+
'…'	vim选项	help 'ruler'
-…	vi命令行选项	help –c

特殊键描述:

- 退格: <BS>
- 回车: <CR> <Enter> <Return>
- 空格/制表(相当于Ctrl-i): <Space> <Tab>
- 箭头: <Up> <Down> <Left> <Right> <Home> <End> <PageUp> <PageDown>
- 功能键: <F1>…<F12>
- Escape键, 相当于CTRL-[: <ESC>
- Ctrl/Shift: <C-…> <S-…>
- Meta/Alt: <M-…> <A-…>

使用help等命令时输入特殊按键

- <C-V>然后按相应的特殊按键

编辑vim配置文件

vim配置文件：~/.vimrc或者~/.vim/vimrc，建议使用后者
mkdir ~/.vim; cd ~/.vim; vi vimrc

- 在正常模式下通过键入iIaAoOcCsS等命令进入插入模式，按Esc键从插入模式切换到正常模式
- 状态栏下方会显示当前模式 --INSERT--　(set showmode)
- 文件内容有改变时状态栏缺省会出现：[+]

vim配置文件
- 采用vim脚本语言编写
- 双引号表示注释
- 命令行模式下执行的命令
- 支持变量赋值let
- 支持if/while/function等程序结构
- :so[urce] vimrc 执行vimrc

```
" 配置文件：~/.vim/vimrc
set nocompatible
set ruler
set showmode
set showcmd
set laststatus=2
set number
set cursorline

syntax enable
filetype plugin indent on

" dark or light
set background=dark   " 或light

colorscheme murphy

set wildmenu
set wildmode=list:longest,full

set mouse=a
```

键	动作
i	当前位置前插入
I	当前行首第一个非空白字符处插入，等价于^i
a	当前位置后插入
A	当前行尾插入，等价于$a
o	当前行下面插入
O	当前行上方插入

选项mouse: 在哪些模式下允许使用鼠标，设置为a，表示所有模式

保存和退出

- **:q[uit]<Return>退出当前窗口，如果为最后一个(帮助等特殊窗口等不计算在内)窗口，则退出vi， Ex命令支持简写，:quit可简写为:q**
  - 许多命令有!版本，表示强制执行该命令
  - **如果有修改但放弃修改，可 :q!<Return>**
- :w[rite]<Return>，保存修改
- :w file<Return>，保存到文件file中，仍然编辑当前文件
- **:wq<Return> 保存修改并退出 :wq!<Return> 强制保存并退出**
- **:w! 强制保存修改**
- :update 有修改才保存
- :x 如果有改动，则保存退出，否则退出

- **键入ZZ 如果有改动，则保存修改并退出，等价于 :x**
- 键入ZQ 不保存修改就退出，等价于:q!

插入模式:终端快捷键和特殊字符

- 插入模式中用户键入的内容会插入到编辑缓冲区中，按Esc键从插入模式切换到正常模式
- vim中还可通过箭头, PageUp, PageDown等键移动光标位置，且仍然在插入模式
- 支持终端快捷键操作
  - Ctrl-H <BS> 删除一个字符　Ctrl-W 删除一个单词　Ctrl-U 删除到行首
  - Ctrl-V然后按特殊字符键或者数字(Unicode码，前缀可为o/x/u/U) 可输入特殊字符(:help i_Ctrl-V)
- 删除时，如果到达行首怎么办?
  - backspace选项如果为空，则无法跨越该行，作用到上一行，作用到进入插入模式之前的位置
  - backspace缺省一般设置为(**set backspace=indent,eol,start**)，indent表示后退删除类操作跨越自动缩进，eol表示跨越end-of-line(删除换行符)，start表示跨越插入模式开始的位置
  - :help fix[del] 如果backspace有问题时
- 输入特殊字符，除了通过Ctrl-V外，还可通过digraph来输入
  - :dig[raphs]列出当前定义的digraph(二合/双并词) **{char1}{char2} char digit**
  - 插入模式下Ctrl-K，然后输入{char1}{char2}，相当于插入char (h i_Ctrl-K)
  - <Ctrl-k>b*可键入β

```
i3 ΐ  912    A* Α  913    B* Β  914    G* Γ  915    D* Δ  916    E* Ε  917
Z* Ζ  918    Y* Η  919    H* Θ  920    I* Ι  921    K* Κ  922    L* Λ  923
M* Μ  924    N* Ν  925    C* Ξ  926    O* Ο  927    P* Π  928    R* Ρ  929
S* Σ  931    T* Τ  932    U* Υ  933    F* Φ  934    X* Χ  935    Q* Ψ  936
W* Ω  937    J* Ϊ  938    V* Ϋ  939    a% ά  940    e% έ  941    y% ή  942
i% ί  943    u3 ΰ  944    a* α  945    b* β  946    g* γ  947    d* δ  948
e* ε  949    z* ζ  950    y* η  951    h* θ  952    i* ι  953    k* κ  954
l* λ  955    m* μ  956    n* ν  957    c* ξ  958    o* ο  959    p* π  960
r* ρ  961    *s ς  962    s* σ  963    t* τ  964    u* υ  965    f* φ  966
x* χ  967    q* ψ  968    w* ω  969    j* ϊ  970    v* ϋ  971    o% ό  972
```

插入模式：缩进

- 缩进：Ctrl-T 当前行再缩进一次， Ctrl-D当前行取消缩进一次，0<Ctrl-D> 删除当前行所有缩进
- autoindent选项(:set autoindent)，即采用自动缩进方式，新的行与上一行为相同的缩进

选项,缩写	含义
tabstop,ts	文件中的Tab字符等同的空格数，缺省=8个空格
softtabstop,sts	插入时按下Tab键，等同于的空格个数。根据ts的设置，实际插入的字符可能为空格与Tab字符的混合
shiftwidth,sw	autoindent(如插入模式下Ctrl-T/Ctrl-D)使用的空格数量，建议与sts一样
expandtab,et	插入模式中将Tab字符转换为空格
autoindent	自动缩进，沿用上一行的缩进
smartindent,si	聪明的自动缩进，识别C语言的部分语法
cindent,cin	采用C程序(Java/C++)所采用的缩进方式 :help indent.txt
indentexpr	根据表达式计算缩进，file plugin indent on时会设置一个合适的表达式

```
set autoindent
set ts=4 sts=4 sw=4
set expandtab
```

- 在设置了expandtab后，要键入Tab：
  <C-V><Tab>

优先级由高到低