

Linux 系统概论

天津医科大学
生物医学工程与技术学院

2017-2018 学年下学期 (春)
2016 级生信班

第七章 Vim 编辑器

伊现富 (Yi Xianfu)

天津医科大学 (TJMU)
生物医学工程与技术学院

2018 年 6 月



教学提纲

1

引言

2

Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3

移动和定位

4

编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5

Vim 进阶

- 运行命令
- Vim 插件

6

Vim 命令集锦

7

Vim 语言

8

回顾与总结

- 总结
- 思考题

教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



引言 | 纯文本 vs. 格式化文本

P8_Aln_Pro - 记事本

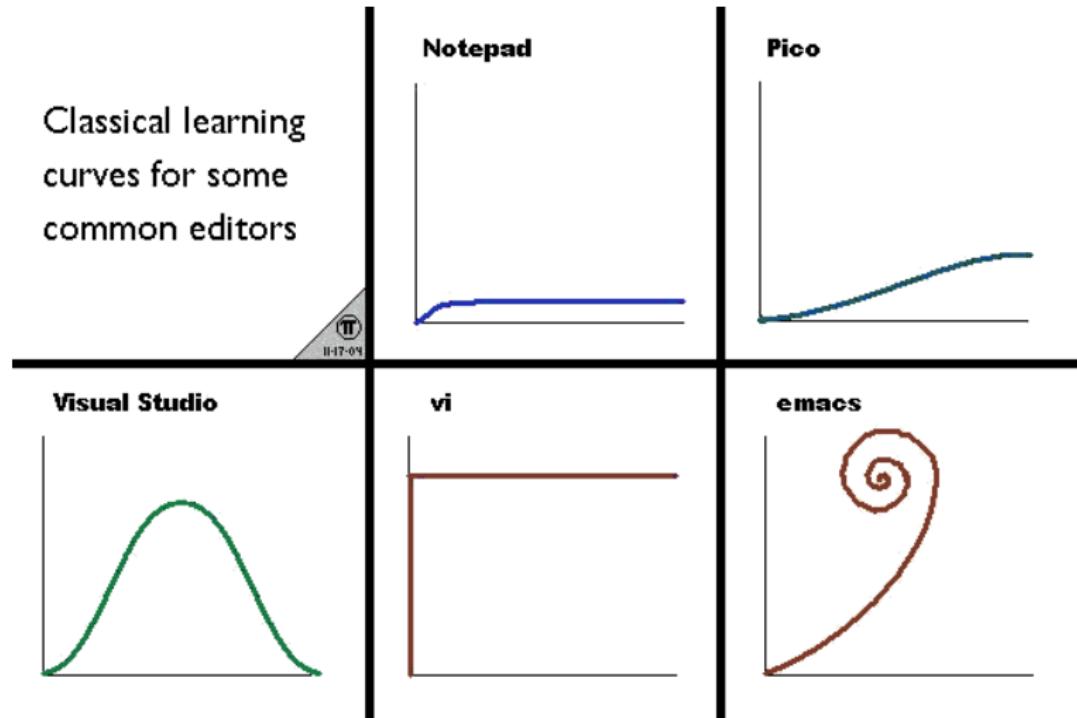
文件(E) 编辑(E) 格式(O) 查看(V) 帮助(H)

CLUSTAL X (1.83) multiple sequence alignment

RGDV_ABC75537 HSRQAWIETSALIECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_AAY14576 HSRQAWIETSALIECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_BAA02676 HSRQAWIETSALIECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_AAY14579 HSRQAWIETSALIECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_AAY14580 HSRQAWIETSALIECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_AA064253 HSRQAWIETSALIERISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_AAY14577 HSRQAWIETSALTECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
RGDV_AAY14578 HSRQAWIETSALIECISEVGTKCSFDTFQLTINDISTLSNLHNQISUASUGFLNDPRTP
WTV_P17380 HSRQNMWVTSALUCECISEVIURSYGDTFGLTSTDLSTSLMSSLMSLISIANUGFLNDRTP
RDVS_Q85451 HSRQNMWLDTSALLEISEVURCNGDFTSGLLTGFDFNALSNNFTQLTSUSSAGYVUDPRUP
RDVO_P17379 HSRQNMWLDTSALLEISEVURCNGDFTSGLLTGFDFNALSNNFTQLTSUSSAGYVUDPRUP
RDVA_Q85449 HSRQNMWLDTSALLEISEVURCNGDFTSGLLTGFDFNALSNNFTQLTSUSSAGYVUDPRUP
RDVF_Q85439 HSRQNMWLDTSALLEISEVURCNGDFTSGLLTGFDFNALSNNFTQLTSUSSAGYVUDPRUP
***** *;***** * * * . . *** *** *.;*****.*;*****.*;*.*

RGDV_ABC75537 LQAHSCEFUNFISTADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU
RGDV_AAY14576 LQAHSCEFUNFISTADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU
RGDV_BAA02676 LQAHSCEFUNFISTADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU
RGDV_AAY14579 LQAHSCEFUNFISTADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU
RGDV_AAY14580 LQAHSCEFUNFISTADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU
RGDV_AA064253 LQAHSCEFUNFISTADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU
RGDV_AAY14577 PQAHSCEFUNFISAADRHYAHYLQKHNFDSDUAPNUTTDNFIAITYIKPFRSRTUSDULRQU





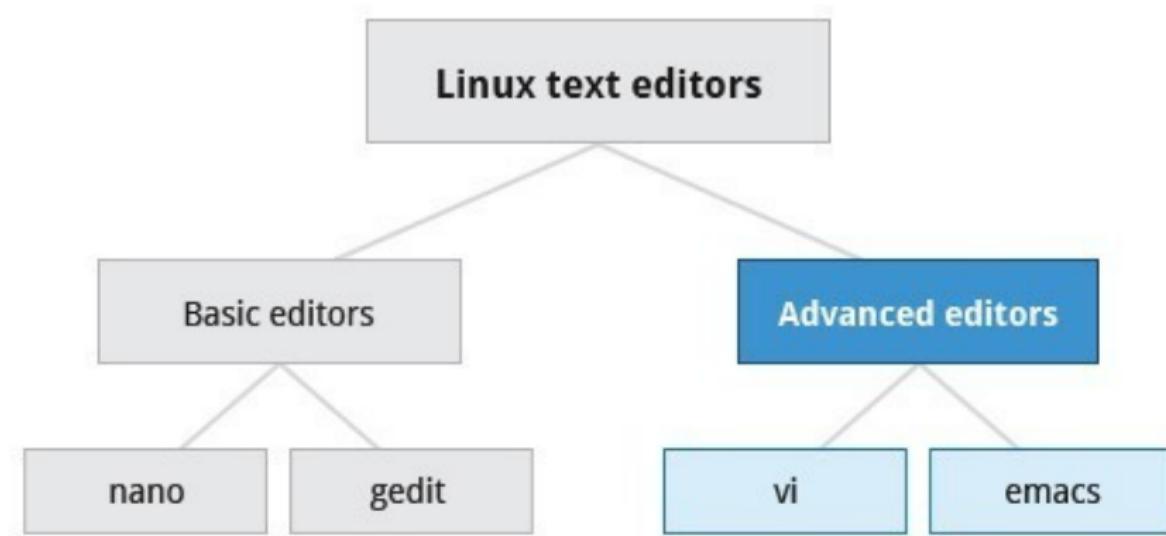
引言 | 文本编辑器

A screenshot of Notepad++ showing a file named index.php. The code is PHP, dealing with session handling and file caching. A code completion dropdown is open at the bottom of the editor window, listing suggestions like 'interface_exists', 'internal', 'is_array', 'is_string', 'is_ip', and 'is_tamed'. The status bar at the bottom shows file length (139209), lines (4673), line number (Ln: 2595), column (Col: 26), scroll position (Sel: 0), file type (UNIX), encoding (ANSI as UTF-8), and insertion mode (INS).

A screenshot of GVim showing the Vim license and help files. The main window displays the Vim license, version 6.0.152, by Bram Moolenaar et al. It mentions that Vim is open source and freely distributable. Below the license, there are sections for 'Help poor children in Uganda!', 'Information', 'On-line help', and 'Version info'. The status bar at the bottom shows line (0,0-1) and column (All).

A screenshot of Sublime Text 2 showing the 'Soda Light.sublime-theme' settings file. The file contains various CSS-like rules for the theme, such as background colors for layers, margins for content, and widths for tabs. A search bar at the top right is set to 'dark'.





三类编辑器

Vim Emacs 其他

大腕版文本编辑器

“周围同事不是用 *Vi* 就是 *Emacs*, 你要是用 *UltraEdit*, 你都不好意思跟人家打招呼……什么插件呀、语法高亮呀、拼写检查呀、能给它开的都给它开着, 就是一个字儿: 酷! 你说这么牛 X 一东西, 怎么着学会也得小半年吧。半年! 入门都远着呢, 能学会移动光标就不错了, 你还别说耗不起, 就这还只是左右移动! ! ”

——《大腕 · 文本编辑器》



三类编辑器

Vim Emacs 其他

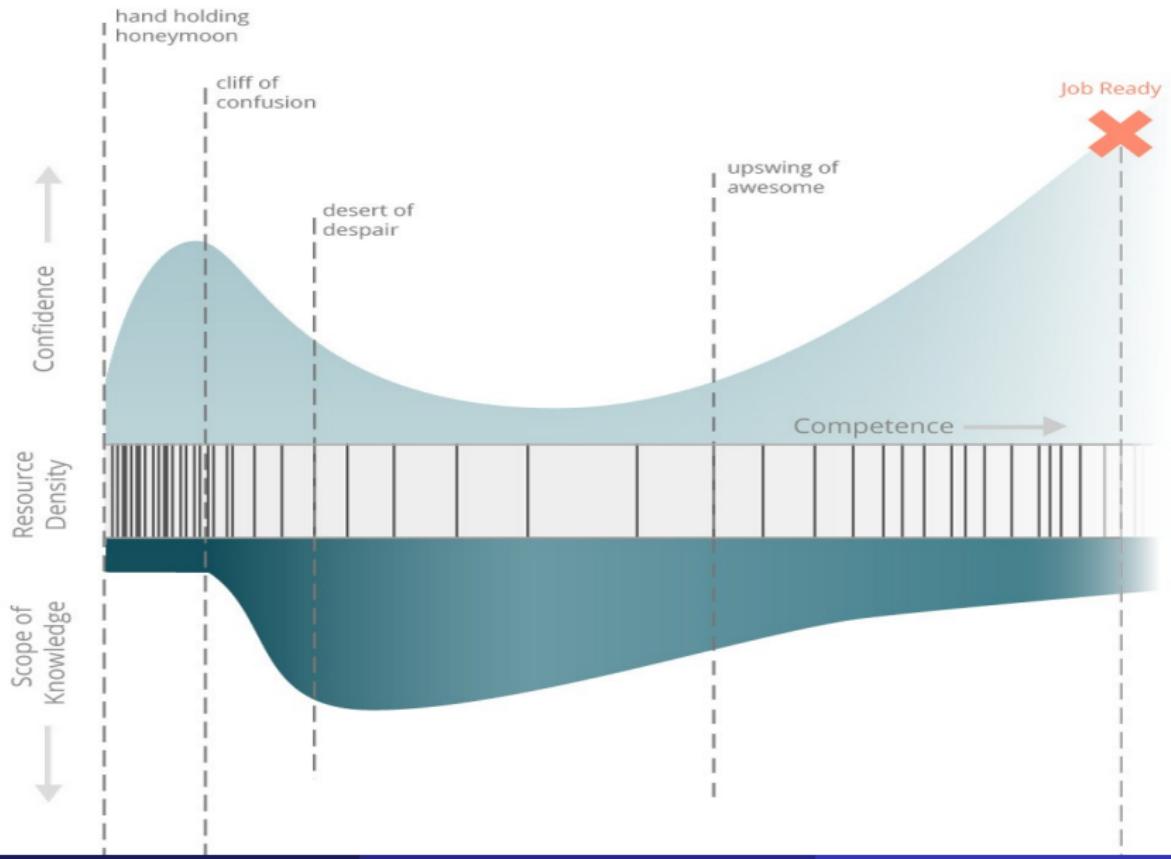
大腕版文本编辑器

“周围同事不是用 *Vi* 就是 *Emacs*, 你要是用 *UltraEdit*, 你都不好意思跟人家打招呼……什么插件呀、语法高亮呀、拼写检查呀、能给它开的都给它开着, 就是一个字儿: 酷! 你说这么牛 X 一东西, 怎么着学会也得小半年吧。半年! 入门都远着呢, 能学会移动光标就不错了, 你还别说耗不起, 就这还只是左右移动! ! ”

——《大腕 · 文本编辑器》



引言 | 文本编辑器 | 学习经历



键盘磨损

- W
- F5
- W、A、S、D
- W、A、S、D、U、I、J、K
- ALT+S 或 CTRL+ENTER
- CTRL+C 和 CTRL+V
- A+SHIFT+CTRL+1+2+3+4+.....
- CTRL+ALT+DEL
- 小键盘
- 以 D 和回车为圆心的两个圆形
- ESC、H、J、K、L
- Ctrl+Alt+Shift
-

用户习惯/职业

- FIFA 高手
- 论坛版主
- 游戏迷 (CS, 35*, ...)
- 拳皇迷
- QQ 狂人
- 网站编辑/转帖狂人
- 星际争霸迷
- 电脑白痴/该换电脑了
- 会计师/银行出纳
- “气管炎”
- Vim
- Emacs
-

键盘磨损

- W
- F5
- W、A、S、D
- W、A、S、D、U、I、J、K
- ALT+S 或 CTRL+ENTER
- CTRL+C 和 CTRL+V
- A+SHIFT+CTRL+1+2+3+4+.....
- CTRL+ALT+DEL
- 小键盘
- 以 D 和回车为圆心的两个圆形
- ESC、H、J、K、L
- Ctrl+Alt+Shift
-

用户习惯/职业

- FIFA 高手
- 论坛版主
- 游戏迷 (CS, 35*, ...)
- 拳皇迷
- QQ 狂人
- 网站编辑/转帖狂人
- 星际争霸迷
- 电脑白痴/该换电脑了
- 会计师/银行出纳
- “气管炎”
- Vim
- Emacs
-

教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



Vim/vi 是一个功能强大的全屏幕文本编辑器，是 Linux/UNIX 上最常用的文本编辑器，它的作用是创建、编辑、显示文本文件。Vim/vi 没有菜单，只有命令。



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



vi 由美国计算机科学家比尔·乔伊 (Bill Joy) 于 1976 年编写、并以 BSD 授权发布的文本编辑器。vi 是 “Visual”的不正规的缩写，来源于另外一个文本编辑器 ex 的命令 visual。

Vim Vi IMproved，从 vi 发展出来的一个文本编辑器，和 Emacs 并列成为类 Unix 系统用户最喜欢的编辑器。布莱姆·米勒在 1991 年发布第一个版本，现在是在开放源代码方式下发行的自由软件。Vim 是 Linux 系统中最常用的 vi 版本，具有 vi 的所有特性，并添加了一些重要的改进，包括语法高亮、代码折叠和多级撤销等。

Vile Vi Like Emacs。试图将 vi 和 Emacs 的优点结合在一起。

Vigor 含有 Vigor 助手的 vi，对抗 Microsoft Office 的 Clippy。

Elvis vi 的升级版本，具有一些额外的特性。

Nvi vi 的 BSD 版本。



vi 由美国计算机科学家比尔·乔伊 (Bill Joy) 于 1976 年编写、并以 BSD 授权发布的文本编辑器。vi 是 “Visual”的不正规的缩写，来源于另外一个文本编辑器 ex 的命令 visual。

Vim Vi IMproved, 从 vi 发展出来的一个文本编辑器，和 Emacs 并列成为类 Unix 系统用户最喜欢的编辑器。布莱姆·米勒在 1991 年发布第一个版本，现在是在开放源代码方式下发行的自由软件。Vim 是 Linux 系统中最常用的 vi 版本，具有 vi 的所有特性，并添加了一些重要的改进，包括语法高亮、代码折叠和多级撤销等。

Vile Vi Like Emacs。试图将 vi 和 Emacs 的优点结合在一起。

Vigor 含有 Vigor 助手的 vi，对抗 Microsoft Office 的 Clippy。

Elvis vi 的升级版本，具有一些额外的特性。

Nvi vi 的 BSD 版本。



vi 由美国计算机科学家比尔·乔伊 (Bill Joy) 于 1976 年编写、并以 BSD 授权发布的文本编辑器。vi 是 “Visual”的不正规的缩写，来源于另外一个文本编辑器 ex 的命令 visual。

Vim Vi IMproved，从 vi 发展出来的一个文本编辑器，和 Emacs 并列成为类 Unix 系统用户最喜欢的编辑器。布莱姆·米勒在 1991 年发布第一个版本，现在是在开放源代码方式下发行的自由软件。Vim 是 Linux 系统中最常用的 vi 版本，具有 vi 的所有特性，并添加了一些重要的改进，包括语法高亮、代码折叠和多级撤销等。

Vile Vi Like Emacs。试图将 vi 和 Emacs 的优点结合在一起。

Vigor 含有 Vigor 助手的 vi，对抗 Microsoft Office 的 Clippy。
Elvis vi 的升级版本，具有一些额外的特性。

Nvi vi 的 BSD 版本。



vi 由美国计算机科学家比尔·乔伊 (Bill Joy) 于 1976 年编写、并以 BSD 授权发布的文本编辑器。vi 是 “Visual”的不正规的缩写，来源于另外一个文本编辑器 ex 的命令 visual。

Vim Vi IMproved, 从 vi 发展出来的一个文本编辑器，和 Emacs 并列成为类 Unix 系统用户最喜欢的编辑器。布莱姆·米勒在 1991 年发布第一个版本，现在是在开放源代码方式下发行的自由软件。Vim 是 Linux 系统中最常用的 vi 版本，具有 vi 的所有特性，并添加了一些重要的改进，包括语法高亮、代码折叠和多级撤销等。

Vile Vi Like Emacs。试图将 vi 和 Emacs 的优点结合在一起。

Vigor 含有 Vigor 助手的 vi，对抗 Microsoft Office 的 Clippy。

Elvis vi 的升级版本，具有一些额外的特性。

Nvi vi 的 BSD 版本。



vi 由美国计算机科学家比尔·乔伊 (Bill Joy) 于 1976 年编写、并以 BSD 授权发布的文本编辑器。vi 是 “Visual”的不正规的缩写，来源于另外一个文本编辑器 ex 的命令 visual。

Vim Vi IMproved，从 vi 发展出来的一个文本编辑器，和 Emacs 并列成为类 Unix 系统用户最喜欢的编辑器。布莱姆·米勒在 1991 年发布第一个版本，现在是在开放源代码方式下发行的自由软件。Vim 是 Linux 系统中最常用的 vi 版本，具有 vi 的所有特性，并添加了一些重要的改进，包括语法高亮、代码折叠和多级撤销等。

Vile Vi Like Emacs。试图将 vi 和 Emacs 的优点结合在一起。

Vigor 含有 Vigor 助手的 vi，对抗 Microsoft Office 的 Clippy。

Elvis vi 的升级版本，具有一些额外的特性。

Nvi vi 的 BSD 版本。



vi 由美国计算机科学家比尔·乔伊 (Bill Joy) 于 1976 年编写、并以 BSD 授权发布的文本编辑器。vi 是 “Visual”的不正规的缩写，来源于另外一个文本编辑器 ex 的命令 visual。

Vim Vi IMproved, 从 vi 发展出来的一个文本编辑器，和 Emacs 并列成为类 Unix 系统用户最喜欢的编辑器。布莱姆·米勒在 1991 年发布第一个版本，现在是在开放源代码方式下发行的自由软件。Vim 是 Linux 系统中最常用的 vi 版本，具有 vi 的所有特性，并添加了一些重要的改进，包括语法高亮、代码折叠和多级撤销等。

Vile Vi Like Emacs。试图将 vi 和 Emacs 的优点结合在一起。

Vigor 含有 Vigor 助手的 vi，对抗 Microsoft Office 的 Clippy。

Elvis vi 的升级版本，具有一些额外的特性。

Nvi vi 的 BSD 版本。



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



入门学习

Vim 初学者教学 在 Linux 系统**命令行下**输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

Vim 帮助系统 在 Vim 中输入



入门学习

Vim 初学者教学 在 Linux 系统**命令行下输入**

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

:help <topic> 或 :h <topic> (topic 可以是命令、模式等)

Vim 帮助系统 在 Vim 中输入



入门学习

Vim 初学者教学 在 Linux 系统命令行下输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

- 英文版： :help user-manual
- 中文版： :help user-manual-zh

Vim 帮助系统 在 Vim 中输入



入门学习

Vim 初学者教学 在 Linux 系统命令行下输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

- 英文版： :help user-manual
- 中文版： :help user-manual@cn

Vim 帮助系统 在 Vim 中输入



入门学习

Vim 初学者教学 在 Linux 系统命令行下输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

- 英文版： :help user-manual
- 中文版： :help user-manual@cn

Vim 帮助系统 在 Vim 中输入



入门学习

Vim 初学者教学 在 Linux 系统命令行下输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

- 英文版： :help user-manual
- 中文版： :help user-manual@cn

Vim 帮助系统 在 Vim 中输入

:help



入门学习

Vim 初学者教学 在 Linux 系统命令行下输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

- 英文版： :help user-manual
- 中文版： :help user-manual@cn

Vim 帮助系统 在 Vim 中输入

- :help



入门学习

Vim 初学者教学 在 Linux 系统命令行下输入

- 英文版： vimtutor
- 中文版： vimtutor -g zh

Vim 用户手册 在 Vim 中输入

- 英文版： :help user-manual
- 中文版： :help user-manual@cn

Vim 帮助系统 在 Vim 中输入

- :help



入门学习

- 对于大多数用户来说，Vim 有着一个比较陡峭的学习曲线。这意味着开始学习的时候可能会进展缓慢，但是一旦掌握一些基本操作之后，能大幅度提高编辑效率。
- 为了帮助学习，Vim 为初学者准备了 Vim 教学。通常可以在 Linux 系统命令行下输入 “vimtutor” 或者点击 Windows 系统桌面上的 Vim 教学图标进入。
- 在 Vim 用户手册中更加详细得描述了 Vim 的基础和进阶功能。可以在 Vim 中输入 “:help user-manual” 进入用户手册。手册除了原始的英文版本之外，也被志愿者翻译成了各国文字，其中包括中文。
- 新用户也应该学习 Vim 的帮助系统，可以在 Vim 中输入不带参数的 “:help” 来阅读主帮助文件。



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



简介 | 启动 | 方法

命令	说明	结果
vim	不使用文件名参数启动 Vim	启动一个空的 Vim 面板。 退出前必须把所做的编辑保存到新文件中
vim filename	使用已有文件名作为参数	在 Vim 中打开已有的文件。保存时将更新文件中修改过的内容
vim filename	使用新文件名作为参数	保存时将使用指定的文件名创建新文件
vim -R filename	以只读模式打开文件	文件是只读的, 不能保存修改 (练习使用 Vim 命令的好方法)
view filename	同 vim -R filename	—
vim -r filename	以修复模式打开文件	—



简介 | 启动 | 界面

```
VIM - Vi IMproved  
版本 7.4  
维护人 Bram Moolenaar 等  
修改者 pkg-vim-maintainers@lists.alioth.debian.org  
Vim 是可自由分发的开放源代码软件  
  
帮助乌干达的可怜儿童！  
输入 :help iccf<Enter>    查看说明  
  
输入 :q<Enter>          退出  
输入 :help<Enter> 或 <F1>  查看在线帮助  
输入 :help version7<Enter>  查看版本信息  
  
NORMAL [未命名]  unix | utf-8 | no ft 100% | 1 0:1
```

```
VIM - Vi IMproved  
version 7.3.154  
by Bram Moolenaar et al.  
Modified by pkg-vim-maintainers@lists.alioth.debian.org  
Vim is open source and freely distributable  
  
Help poor children in Uganda!  
type :help iccf<Enter>      for information  
  
type :q<Enter>          to exit  
type :help<Enter> or <F1>  for on-line help  
type :help version7<Enter>  for version info  
  
0,0-1  All
```

- 行首的颤化符号 (~) 表示未使用的行。不以~开始却仍然是空白的，说明存在空格、制表符或其他不可见字符。
- 屏幕底部是状态行，显示文件名和光标所在的行等信息。
- 打开文件时，拒绝访问或显示只读，可能存在权限问题。
- 打开非文本文件可能会显示乱码，输入 :q! 直接退出即可。
- 启动 Vim 后默认进入命令模式。要输入文本，必须进入输入模式。



- 行首的颤化符号 (~) 表示未使用的行。不以~开始却仍然是空白的，说明存在空格、制表符或其他不可见字符。
- 屏幕底部是状态行，显示文件名和光标所在的行等信息。
- 打开文件时，拒绝访问或显示只读，可能存在权限问题。
- 打开非文本文件可能会显示乱码，输入 :q! 直接退出即可。
- 启动 Vim 后默认进入命令模式。要输入文本，必须进入输入模式。



- 行首的颤化符号 (~) 表示未使用的行。不以~开始却仍然是空白的，说明存在空格、制表符或其他不可见字符。
- 屏幕底部是状态行，显示文件名和光标所在的行等信息。
- 打开文件时，拒绝访问或显示只读，可能存在权限问题。
- 打开非文本文件可能会显示乱码，输入 :q! 直接退出即可。
- 启动 Vim 后默认进入命令模式。要输入文本，必须进入输入模式。



- 行首的颤化符号 (~) 表示未使用的行。不以~开始却仍然是空白的，说明存在空格、制表符或其他不可见字符。
- 屏幕底部是状态行，显示文件名和光标所在的行等信息。
- 打开文件时，拒绝访问或显示只读，可能存在权限问题。
- 打开非文本文件可能会显示乱码，输入 :q! 直接退出即可。
- 启动 Vim 后默认进入命令模式。要输入文本，必须进入输入模式。



- 行首的颤化符号 (~) 表示未使用的行。不以~开始却仍然是空白的，说明存在空格、制表符或其他不可见字符。
- 屏幕底部是状态行，显示文件名和光标所在的行等信息。
- 打开文件时，拒绝访问或显示只读，可能存在权限问题。
- 打开非文本文件可能会显示乱码，输入 :q! 直接退出即可。
- 启动 Vim 后默认进入命令模式。要输入文本，必须进入输入模式。



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



简介 | 状态行

文件名 类型 编码 光标处字符 行,列,百分比 时间日期星期
ASCII码的10,16进

The screenshot shows the Vim status bar with several components labeled:

- Mode Indicator: NORMAL | GVIM1
- File Path: ~/Documents/projects/vim-neatstatus/plugin/neatstatus.vim
- File Type: vim
- File Encoding: unix
- File Format: latin1
- Buffer Number: BUF #1
- Cursor Position: LN 105/143 | (73%) | COL 0-1

The screenshot shows the Vim status bar with several components labeled:

- current line / total lines : column
- encoding
- scroll %
- trailing spaces
- Syntastic.vim errors

Annotations point to specific fields in the status bar:

- buffer number
- file name
- modified
- filetype
- moveRoute
- i = arrayIndex0f(this._routes, route);
- L73/357:C23
- 18% [Syntax: line:76 (1)][\s]

教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



命令模式 (Command Mode) Vim 启动后的默认模式，所输入的任何内容都被解释成命令。在此模式中，用户可以执行一般的编辑器命令，比如移动光标、删除文本等等。在命令模式中，有很多方法可以进入输入模式，比较普通的方式是按 “a” (append, 追加) 键或者 “i” (insert, 插入) 键。

输入模式 (Insert Mode) 在此模式中，大多数按键都会向文本缓冲中插入文本，可以按 ESC 键回到命令模式。

末行模式 (Last Line Mode) 在末行模式中可以输入会被解释并执行的文本。例如执行命令 (“:” 键)，搜索 (“/” 和 “?” 键) 或者过滤命令 (“!” 键)。在命令执行之后，Vim 返回到末行模式之前的模式，通常是普通模式（命令模式）。



命令模式 (Command Mode) Vim 启动后的默认模式，所输入的任何内容都被解释成命令。在此模式中，用户可以执行一般的编辑器命令，比如移动光标、删除文本等等。在命令模式中，有很多方法可以进入输入模式，比较普通的方式是按 “a” (append, 追加) 键或者 “i” (insert, 插入) 键。

输入模式 (Insert Mode) 在此模式中，大多数按键都会向文本缓冲中插入文本，可以按 ESC 键回到命令模式。

末行模式 (Last Line Mode) 在末行模式中可以输入会被解释并执行的文本。例如执行命令 (“:” 键)，搜索 (“/” 和 “?” 键) 或者过滤命令 (“!” 键)。在命令执行之后，Vim 返回到末行模式之前的模式，通常是普通模式（命令模式）。



命令模式 (Command Mode) Vim 启动后的默认模式，所输入的任何内容都被解释成命令。在此模式中，用户可以执行一般的编辑器命令，比如移动光标、删除文本等等。在命令模式中，有很多方法可以进入输入模式，比较普通的方式是按 “a” (append, 追加) 键或者 “i” (insert, 插入) 键。

输入模式 (Insert Mode) 在此模式中，大多数按键都会向文本缓冲中插入文本，可以按 ESC 键回到命令模式。

末行模式 (Last Line Mode) 在末行模式中可以输入会被解释并执行的文本。例如执行命令 (“:” 键)，搜索 (“/” 和 “?” 键) 或者过滤命令 (“!” 键)。在命令执行之后，Vim 返回到末行模式之前的模式，通常是普通模式（命令模式）。



可视模式 与命令模式比较相似，但是移动命令会扩大高亮的文本区域。高亮区域可以是字符、行或者是一块文本。当执行一个非移动命令时，命令会被执行到这块高亮的区域上。

替换模式 一个特殊的插入模式，在这个模式中可以做和插入模式一样的操作，但是每个输入的字符都会覆盖文本缓冲中已经存在的字符。在命令模式下按“R”键进入。

选择模式 和无模式编辑器的行为比较相似。这个模式中，可以用鼠标或者光标键高亮选择文本，不过输入任何字符的话，Vim 会用这个字符替换选择的高亮文本块，并且自动进入输入模式。

Ex 模式 和命令行模式比较相似，在使用“:visual”命令离开 Ex 模式前，可以一次执行多条命令。在命令模式下按“Q”键进入。



可视模式 与命令模式比较相似，但是移动命令会扩大高亮的文本区域。高亮区域可以是字符、行或者是一块文本。当执行一个非移动命令时，命令会被执行到这块高亮的区域上。

替换模式 一个特殊的插入模式，在这个模式中可以做和插入模式一样的操作，但是每个输入的字符都会覆盖文本缓冲中已经存在的字符。在命令模式下按“R”键进入。

选择模式 和无模式编辑器的行为比较相似。这个模式中，可以用鼠标或者光标键高亮选择文本，不过输入任何字符的话，Vim 会用这个字符替换选择的高亮文本块，并且自动进入输入模式。

Ex 模式 和命令行模式比较相似，在使用“:visual”命令离开 Ex 模式前，可以一次执行多条命令。在命令模式下按“Q”键进入。



可视模式 与命令模式比较相似，但是移动命令会扩大高亮的文本区域。高亮区域可以是字符、行或者是一块文本。当执行一个非移动命令时，命令会被执行到这块高亮的区域上。

替换模式 一个特殊的插入模式，在这个模式中可以做和插入模式一样的操作，但是每个输入的字符都会覆盖文本缓冲中已经存在的字符。在命令模式下按“R”键进入。

选择模式 和无模式编辑器的行为比较相似。这个模式中，可以用鼠标或者光标键高亮选择文本，不过输入任何字符的话，Vim 会用这个字符替换选择的高亮文本块，并且自动进入输入模式。

Ex 模式 和命令行模式比较相似，在使用“:visual”命令离开 Ex 模式前，可以一次执行多条命令。在命令模式下按“Q”键进入。



可视模式 与命令模式比较相似，但是移动命令会扩大高亮的文本区域。高亮区域可以是字符、行或者是一块文本。当执行一个非移动命令时，命令会被执行到这块高亮的区域上。

替换模式 一个特殊的插入模式，在这个模式中可以做和插入模式一样的操作，但是每个输入的字符都会覆盖文本缓冲中已经存在的字符。在命令模式下按“R”键进入。

选择模式 和无模式编辑器的行为比较相似。这个模式中，可以用鼠标或者光标键高亮选择文本，不过输入任何字符的话，Vim 会用这个字符替换选择的高亮文本块，并且自动进入输入模式。

Ex 模式 和命令行模式比较相似，在使用“:visual”命令离开 Ex 模式前，可以一次执行多条命令。在命令模式下按“Q”键进入。



简介 | 工作模式 | 模式转换

命令模式下按“I”、“o”、“a”键或“Insert”键就可以切换到输入模式，该模式中的主要操作就是录入文件内容，可以对文件正文进行修改、或者添加新的内容。处于输入模式时，vi编辑器的最后一行会出现“—INSERT—”的状态提示信息。

输入模式

输入
[Insert]、
[a]、[i]
[o]

输入[Esc]

[root@localhost ~]# vim file_name

命令模式

输入冒号[:]

输入[Esc]

末行模式

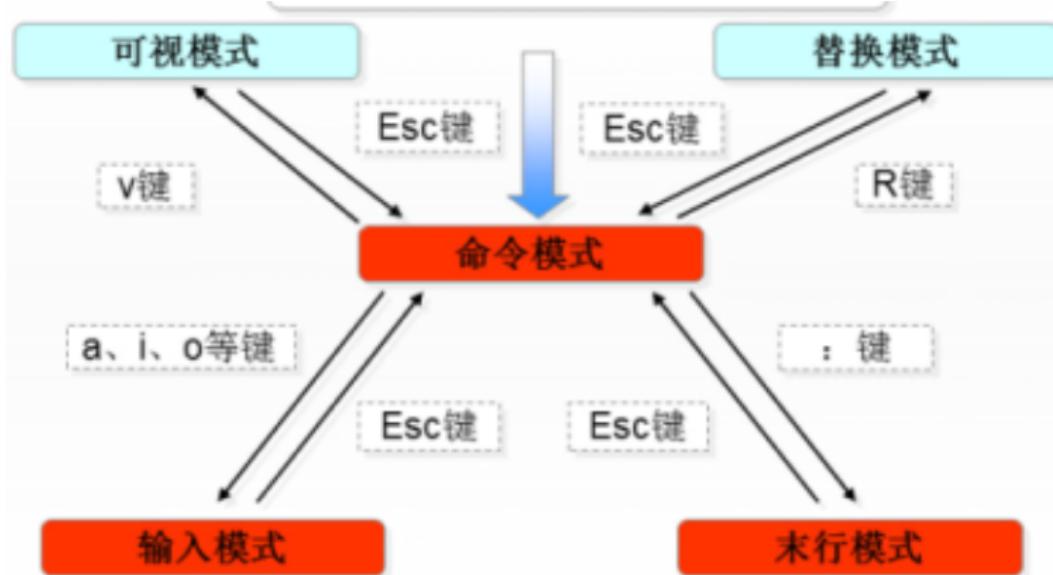
启动vi编辑器后默认进入命令模式，该模式下主要完成如光标移动、字符串查找、删除、复制、粘贴等操作。不论用户处于何种模式，只要按下Esc键，即可进入命令模式。

在命令模式下，按“:”键即可进入末行模式，该模式中可以保存文件、退出编辑器，以及对文件内容进行查找、替换等操作。

处于末行模式时，vi编辑器的最后一行会出现“:”提示符



简介 | 工作模式 | 模式转换



- 注意状态栏中显示的工作模式及只读指示。
- 一般按 Esc 键即可退出其他模式。
- 可以按两次 Esc 键以确保回到命令模式。
- Vim 命令区分大小写。
- 接下来介绍的命令绝大多数都基于命令模式。

- 注意状态栏中显示的工作模式及只读指示。
- 一般按 Esc 键即可退出其他模式。
- 可以按两次 Esc 键以确保回到命令模式。
- Vim 命令区分大小写。
- 接下来介绍的命令绝大多数都基于命令模式。

- 注意状态栏中显示的工作模式及只读指示。
- 一般按 Esc 键即可退出其他模式。
- 可以按两次 Esc 键以确保回到命令模式。
- Vim 命令区分大小写。
- 接下来介绍的命令绝大多数都基于命令模式。

- 注意状态栏中显示的工作模式及只读指示。
- 一般按 Esc 键即可退出其他模式。
- 可以按两次 Esc 键以确保回到命令模式。
- Vim 命令区分大小写。
- 接下来介绍的命令绝大多数都基于命令模式。



- 注意状态栏中显示的工作模式及只读指示。
- 一般按 Esc 键即可退出其他模式。
- 可以按两次 Esc 键以确保回到命令模式。
- Vim 命令区分大小写。
- 接下来介绍的命令绝大多数都基于命令模式。



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

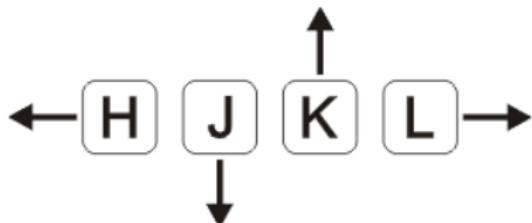
- 回顾与总结
- 总结
- 思考题



移动和定位 | 基本移动



命令	说明	结果
h	小写 H	左移一格 (Backspace)
j	小写 J	下移一行
k	小写 K	上移一行
l	小写 L	右移一格 (Space)
10j	hjkl 前加数字	下移 10 行
6h	hjkl 前加数字	左移 6 个字符



移动和定位 | 移动定位

命令	说明	结果
0	零	光标移至行首
^	—	光标移至行首（第一个非空字符）
\$	美元符号	光标移至行尾
b	小写 b	光标移至当前单词或前一个单词的开头
B	大写 B	同 b, 但不计算标点符号
w	小写 w	光标移至下一个单词的开头
W	大写 W	同 w, 但不计算标点符号
e	小写 e	光标移至当前单词或下一个单词的末尾
E	大写 E	同 e, 但不计算标点符号
ge	—	光标移至上一个单词的末尾
gE	—	同 ge, 但不计算标点符号



移动和定位 | 移动定位

命令	说明	结果
	管道符	光标移至行首
n	—	光标移至当前行的第 n 列
_	下划线	光标移至行首（第一个非空字符，支持向下计数）
+	加号	光标移至下一行行首
-	减号	光标移至上一行行首
gg	两个小写 g	光标移至第一行 (:0)
G	大写 G	光标移至最后一行 (:\$)
nG	G 前加数字	光标移至指定行
:n	冒号后跟数字	光标移至指定行
(左圆括号	光标移至当前句子或上一句子的开始处
)	右圆括号	光标移至下一句子的开始处
{	左大括号	光标移至段落的开始处
}	右大括号	光标移至段落的结尾处



移动和定位 | 移动定位

命令	说明	结果
*	星字符	向下查找光标下的单词
#	井字符	向上查找光标下的单词
%	大中小括号	查找与光标下括号相配对的括号
fx	向下查找	光标移至下一个 x 上
Fx	向上查找	光标移至上一个 x 上
;	分号	同向查找
,	逗号	反向查找
tx	—	光标移至下一个 x 之前
Tx	—	光标移至上一个 x 之后



移动和定位 | 移动定位

命令	说明	结果
''	两个单引号	光标移至上一个编辑位置 (所在行首个非空白字符)
'.	单引号加点号	同 ''
``	两个反引号	光标移至上一个编辑位置 (精确到列)
`.	反引号加点号	同 ``
Ctrl+O	—	光标跳转至更老的编辑位置
Ctrl+I	—	光标跳转至更新的编辑位置
mx	—	设置名为 x 的标记 (mark)
' x	—	光标跳转至标记 x (所在行首个非空白字符)
` x	—	光标跳转至标记 x (精确到列)



移动和定位 | 移动定位

命令	说明	结果
H	大写 H	光标移至屏幕上端
M	大写 M	光标移至屏幕中央
L	大写 L	光标移至屏幕下端
zt	—	把光标所在行置于屏幕顶部
zz	—	把光标所在行置于屏幕中部
zb	—	把光标所在行置于屏幕底部
Ctrl+E	同时按下 Ctrl 和 E 键	屏幕上滚
Ctrl+Y	同时按下 Ctrl 和 Y 键	屏幕下滚
Ctrl+F	同时按下 Ctrl 和 F 键	向下滚动一屏
Ctrl+B	同时按下 Ctrl 和 B 键	向上滚动一屏
Ctrl+D	同时按下 Ctrl 和 D 键	向下滚动半屏
Ctrl+U	同时按下 Ctrl 和 U 键	向上滚动半屏
Ctrl+G	同时按下 Ctrl 和 G 键	显示状态，确定光标位置

移动和定位 | 移动定位 | 说明

命令	说明	结果
:set nu	末行模式	显示行号
:set nonu	末行模式	隐藏行号
:set ic	末行模式 (ignorecase)	查找时忽略字母大小写
:set is	末行模式 (incsearch)	查找短语时显示部分匹配
:set hls	末行模式 (hlsearch)	高亮显示所有的匹配短语
—	单词	一组由空格、Tab 或标点符号分隔开的字符
—	句子	由后面跟两个空格的标点符号标识



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



命令	说明	结果
i	小写 i	insert, 在光标前插入
I	大写 I	在行首处插入
a	小写 a	append, 在光标后插入
A	大写 A	在行尾处插入
o	小写 o	在光标所在行的下面插入新行
O	大写 O	在光标所在行的上面插入新行
2O	命令前加数字	在光标所在行的上面插入两个新行



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



命令	说明	结果
x	小写 x	删除光标所在位置的字符
X	大写 X	删除光标前的字符
dw	小写 dw	从光标处删除至下一个单词
d^	—	从光标前删除至行首
d\$	—	从光标处删除至行尾
dG	—	从光标所在行删除至最后一行
D	大写 D	从光标处删除至行尾
dd	小写 dd	删除光标所在行
3x	命令前加数字	删除光标所在位置开始的后 3 个字符
3dd, d3d	命令前加数字	删除 3 行
:m, nd	—	从第 m 行删除到第 n 行
ddO	组合命令	删除光标所在行并创建新行

命令	说明	结果
di (—	删除引号、括号等内部的文字	
da (—	删除引号、括号等及其内部的文字	
dtx —	删除至 x (删除 x 前的内容)	



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



命令	说明	结果
cc	两个小写 c	修改光标所在行
cw	小写 cw	修改光标所在的单词（光标处至单词结束），进入输入模式
r	小写 r	替换光标处的字符，替换后返回命令模式
R	大写 R	从光标处开始替换字符，直至按 Esc 停止
s	小写 s	替换光标处的字符，替换后处于输入模式
S	大写 S	替换光标所在行，替换后处于输入模式



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

● 复制粘贴

- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



命令	说明	结果
J	大写 J	合并光标所在行和其下一行
yy	两个小写 y	复制光标所在行
Y	大写 Y	复制光标所在行
yw	小写 yw	复制光标所在的单词（光标处至单词结尾）
p	小写 p	在光标后粘贴
P	大写 P	在光标前粘贴
3yy, y3y	命令前加数字	复制从光标所在行开始的 3 行内容



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

● 复制粘贴

● 撤销重做

● 搜索替换

● 保存退出

● 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

● 总结

● 思考题



编辑 | 撤销重做

命令	说明	结果
u	小写 u	撤销最近一次的编辑
U	大写 U	撤销光标所在行的所有编辑
Ctrl+R	同时按下 Ctrl 和 R 键	重做
.	句点	重复上一个命令



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- **搜索替换**
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



命令	结果
/word	从光标处向下搜索 word
?word	从光标处向上搜索 word
n	同向定位下一个搜索
N	反向定位下一个搜索



命令	结果
:s/old/new/	将光标所在行上的第一个 old 替换为 new
:s/old/new/g	globally, 将光标所在行的所有 old 替换为 new
:s/old/new/c	替换时进行确认
:m,ns/old/new/g	将从第 m 行到第 n 行的所有 old 替换为 new
:1,\$s/old/new/g	将整个文件内的所有 old 替换为 new
:%s/old/new/g	将整个文件内的所有 old 替换为 new
:%s/\<old\>/new/g	将整个文件内的所有 old 单词替换为 new



命令	结果
默认	对光标所在行进行操作
m, n	在从第 m 行到第 n 行范围内进行操作
%	在整个文件内进行操作
/g	globally, 对所有匹配进行替换
/c	每次替换前进行询问确认
:m, ns/^/#/g	进行连续行注释
:m, ns/^#/g	删除连续行注释
&	重复最后一个替换命令 (:s)



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- **保存退出**
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



保存退出

命令	结果
:q	未修改，退出 Vim 编辑器
:w	写入修改，保存当前文件
:w filename	文件另存为 filename
:wq	保存文件并退出
ZZ	同 :wq
:x	保存文件并退出
:q!	放弃对文件的修改并退出
:w! filename	覆盖文件
:wq!	保存修改并退出（文件所有者可忽略文件的只读属性）
:e!	打开文件的上一次成功写入的版本
:r /path/filename	插入 filename 文件的内容



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



命令	结果
v	进入可视模式
V	进入行可视模式
Ctrl+V	进入块/列可视模式
~	转换光标下字符的大小写
>	可视模式, 缩进
>>	行首缩进
<	可视模式, 反向缩进
<<	行首反向缩进
=	可视模式, 自动缩进/格式化
==	自动缩进
qx	录制宏, 命名为 x, 按 q 键结束录制
@x	运行名为 x 的宏
K	查找光标下单词的 man page

命令补遗

命令	结果
:split	横向分割窗口
:vsplit	纵向分割窗口
:diffsplit	以横向比较模式分割窗口
:vert diffsplit	以纵向比较模式分割窗口
Ctrl+W Ctrl+W	在窗口间跳转
Ctrl+W p	在窗口间跳转（下一个窗口）
Ctrl+W h/j/k/l	在窗口间跳转



命令补遗

命令	结果
缩写	:ab asap as soon as possible
:w !sudo tee %	忘记用 root 打开文件时的文件保存
:earlier 1m	按时间回退文件
:later	与上述相反
基本计算器	插入模式下，依次：Ctrl+R, =, 算式, Enter
:TOhtml	把当前文件转换成网页
:help	打开帮助窗口
:help CMD	找到关于 CMD 命令的帮助信息
Ctrl+N	输入模式，自动补全
Ctrl+D	输入 : 命令时，查看可能的补全结果
Tab	输入 : 命令时自动补全
vimdiff	比较两个文件的不同（在终端中使用）
~/.vimrc	使用 vimrc 启动脚本文件保存用户偏好设置
更多	等待你去发现

教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5

Vim 进阶

- 运行命令
- Vim 插件

6

Vim 命令集锦

7

Vim 语言

8

回顾与总结

- 总结
- 思考题



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



命令	结果
:!command	在 Vim 中运行命令，按任意键返回 Vim
:!ls	在 Vim 中运行 ls 命令
:!wc %	在 Vim 中运行 wc 命令（% 代表当前文件）
:r !command	插入命令执行结果
:r !date	插入日期和时间
:r !sed -n 1,3p filename	插入 filename 文件的 1-3 行内容



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



进阶 | Vim 插件

```
[1:main.c]*+[10:macros.h][11:eval.c][12:misc2.c] 1 [Line:1/1,Column:1] [100%]
[-miniBufExplorer-] [-]

4  /* /home/wooin/vim71/src/
5  ...
6  auto/
7  objects/
8  proto/
9  testdir/
10 xxd/
11 INSTALL
12 Makefile
13 README.txt
14 arabic.c
15 buffer.c
16 charset.c
17 config.aap.in
18 config.h.in
19 config.mk.dist
20 config.mk.in
[<-][Line:5/141,Column:1] [3%]

 3
1400 .      disallow_gui = TRUE;
1401
1402     /* TODO: On MacOS X default to gui if argv[0] ends in:
1403     *          /Vim.app/Contents/MacOS/Vim */
1404 #endif
1405
1406 #ifdef FEAT_EVAL
1407     set_vim_var_string(VV_PROGNAME, initstr, -1);
1408 #endif
1409
1410     if (TOLOWER_ASC(initstr[0]) == 'r')
1411     {
1412         restricted = TRUE;
1413         ++initstr;
1414     }
1415
1416     /* Avoid using evim mode for "editor". */
1417     if (TOLOWER_ASC(initstr[0]) == 'e'
1418     && (TOLOWER_ASC(initstr[1]) == 'v'
1419     .     || TOLOWER_ASC(initstr[1]) == 'g'))
1420     {
1421 #ifdef FEAT_GUI
1422         gui.starting = TRUE;
1423 #endif
1424         parmp->evim_mode = TRUE;
1425         ++initstr
1426     }
1427     if (TOLOWE
1428     {
1429         main_s
1430         main_
1431 #ifdef FEAT_GU
1432         +initstr
1433 #endif
1434     }
1435
1436     if (TOLOWE
1437     {
1438         main_s
1439         main_
1440         main_
1441 #ifdef FEAT_GU
1442         +initstr
1443 #endif
1444     }
1445
1446     if (TOLOWE
1447     {
1448         main_s
1449         main_
1450         main_
1451 #ifdef FEAT_GU
1452         +initstr
1453 #endif
1454     }
1455
1456     if (TOLOWE
1457     {
1458         main_s
1459         main_
1460         main_
1461 #ifdef FEAT_GU
1462         +initstr
1463 #endif
1464     }
1465
1466     if (TOLOWE
1467     {
1468         main_s
1469         main_
1470         main_
1471 #ifdef FEAT_GU
1472         +initstr
1473 #endif
1474     }
1475
1476     if (TOLOWE
1477     {
1478         main_s
1479         main_
1480         main_
1481 #ifdef FEAT_GU
1482         +initstr
1483 #endif
1484     }
1485
1486     if (TOLOWE
1487     {
1488         main_s
1489         main_
1490         main_
1491 #ifdef FEAT_GU
1492         +initstr
1493 #endif
1494     }
1495
1496     if (TOLOWE
1497     {
1498         main_s
1499         main_
1500         main_
1501 #ifdef FEAT_GU
1502         +initstr
1503 #endif
1504     }
1505
1506     if (TOLOWE
1507     {
1508         main_s
1509         main_
1510         main_
1511 #ifdef FEAT_GU
1512         +initstr
1513 #endif
1514     }
1515
1516     if (TOLOWE
1517     {
1518         main_s
1519         main_
1520         main_
1521 #ifdef FEAT_GU
1522         +initstr
1523 #endif
1524     }
1525
1526     if (TOLOWE
1527     {
1528         main_s
1529         main_
1530         main_
1531 #ifdef FEAT_GU
1532         +initstr
1533 #endif
1534     }
1535
1536     if (TOLOWE
1537     {
1538         main_s
1539         main_
1540         main_
1541 #ifdef FEAT_GU
1542         +initstr
1543 #endif
1544     }
1545
1546     if (TOLOWE
1547     {
1548         main_s
1549         main_
1550         main_
1551 #ifdef FEAT_GU
1552         +initstr
1553 #endif
1554     }
1555
1556     if (TOLOWE
1557     {
1558         main_s
1559         main_
1560         main_
1561 #ifdef FEAT_GU
1562         +initstr
1563 #endif
1564     }
1565
1566     if (TOLOWE
1567     {
1568         main_s
1569         main_
1570         main_
1571 #ifdef FEAT_GU
1572         +initstr
1573 #endif
1574     }
1575
1576     if (TOLOWE
1577     {
1578         main_s
1579         main_
1580         main_
1581 #ifdef FEAT_GU
1582         +initstr
1583 #endif
1584     }
1585
1586     if (TOLOWE
1587     {
1588         main_s
1589         main_
1590         main_
1591 #ifdef FEAT_GU
1592         +initstr
1593 #endif
1594     }
1595
1596     if (TOLOWE
1597     {
1598         main_s
1599         main_
1600         main_
1601 #ifdef FEAT_GU
1602         +initstr
1603 #endif
1604     }
1605
1606     if (TOLOWE
1607     {
1608         main_s
1609         main_
1610         main_
1611 #ifdef FEAT_GU
1612         +initstr
1613 #endif
1614     }
1615
1616     if (TOLOWE
1617     {
1618         main_s
1619         main_
1620         main_
1621 #ifdef FEAT_GU
1622         +initstr
1623 #endif
1624     }
1625
1626     if (TOLOWE
1627     {
1628         main_s
1629         main_
1630         main_
1631 #ifdef FEAT_GU
1632         +initstr
1633 #endif
1634     }
1635
1636     if (TOLOWE
1637     {
1638         main_s
1639         main_
1640         main_
1641 #ifdef FEAT_GU
1642         +initstr
1643 #endif
1644     }
1645
1646     if (TOLOWE
1647     {
1648         main_s
1649         main_
1650         main_
1651 #ifdef FEAT_GU
1652         +initstr
1653 #endif
1654     }
1655
1656     if (TOLOWE
1657     {
1658         main_s
1659         main_
1660         main_
1661 #ifdef FEAT_GU
1662         +initstr
1663 #endif
1664     }
1665
1666     if (TOLOWE
1667     {
1668         main_s
1669         main_
1670         main_
1671 #ifdef FEAT_GU
1672         +initstr
1673 #endif
1674     }
1675
1676     if (TOLOWE
1677     {
1678         main_s
1679         main_
1680         main_
1681 #ifdef FEAT_GU
1682         +initstr
1683 #endif
1684     }
1685
1686     if (TOLOWE
1687     {
1688         main_s
1689         main_
1690         main_
1691 #ifdef FEAT_GU
1692         +initstr
1693 #endif
1694     }
1695
1696     if (TOLOWE
1697     {
1698         main_s
1699         main_
1700         main_
1701 #ifdef FEAT_GU
1702         +initstr
1703 #endif
1704     }
1705
1706     if (TOLOWE
1707     {
1708         main_s
1709         main_
1710         main_
1711 #ifdef FEAT_GU
1712         +initstr
1713 #endif
1714     }
1715
1716     if (TOLOWE
1717     {
1718         main_s
1719         main_
1720         main_
1721 #ifdef FEAT_GU
1722         +initstr
1723 #endif
1724     }
1725
1726     if (TOLOWE
1727     {
1728         main_s
1729         main_
1730         main_
1731 #ifdef FEAT_GU
1732         +initstr
1733 #endif
1734     }
1735
1736     if (TOLOWE
1737     {
1738         main_s
1739         main_
1740         main_
1741 #ifdef FEAT_GU
1742         +initstr
1743 #endif
1744     }
1745
1746     if (TOLOWE
1747     {
1748         main_s
1749         main_
1750         main_
1751 #ifdef FEAT_GU
1752         +initstr
1753 #endif
1754     }
1755
1756     if (TOLOWE
1757     {
1758         main_s
1759         main_
1760         main_
1761 #ifdef FEAT_GU
1762         +initstr
1763 #endif
1764     }
1765
1766     if (TOLOWE
1767     {
1768         main_s
1769         main_
1770         main_
1771 #ifdef FEAT_GU
1772         +initstr
1773 #endif
1774     }
1775
1776     if (TOLOWE
1777     {
1778         main_s
1779         main_
1780         main_
1781 #ifdef FEAT_GU
1782         +initstr
1783 #endif
1784     }
1785
1786     if (TOLOWE
1787     {
1788         main_s
1789         main_
1790         main_
1791 #ifdef FEAT_GU
1792         +initstr
1793 #endif
1794     }
1795
1796     if (TOLOWE
1797     {
1798         main_s
1799         main_
1800         main_
1801 #ifdef FEAT_GU
1802         +initstr
1803 #endif
1804     }
1805
1806     if (TOLOWE
1807     {
1808         main_s
1809         main_
1810         main_
1811 #ifdef FEAT_GU
1812         +initstr
1813 #endif
1814     }
1815
1816     if (TOLOWE
1817     {
1818         main_s
1819         main_
1820         main_
1821 #ifdef FEAT_GU
1822         +initstr
1823 #endif
1824     }
1825
1826     if (TOLOWE
1827     {
1828         main_s
1829         main_
1830         main_
1831 #ifdef FEAT_GU
1832         +initstr
1833 #endif
1834     }
1835
1836     if (TOLOWE
1837     {
1838         main_s
1839         main_
1840         main_
1841 #ifdef FEAT_GU
1842         +initstr
1843 #endif
1844     }
1845
1846     if (TOLOWE
1847     {
1848         main_s
1849         main_
1850         main_
1851 #ifdef FEAT_GU
1852         +initstr
1853 #endif
1854     }
1855
1856     if (TOLOWE
1857     {
1858         main_s
1859         main_
1860         main_
1861 #ifdef FEAT_GU
1862         +initstr
1863 #endif
1864     }
1865
1866     if (TOLOWE
1867     {
1868         main_s
1869         main_
1870         main_
1871 #ifdef FEAT_GU
1872         +initstr
1873 #endif
1874     }
1875
1876     if (TOLOWE
1877     {
1878         main_s
1879         main_
1880         main_
1881 #ifdef FEAT_GU
1882         +initstr
1883 #endif
1884     }
1885
1886     if (TOLOWE
1887     {
1888         main_s
1889         main_
1890         main_
1891 #ifdef FEAT_GU
1892         +initstr
1893 #endif
1894     }
1895
1896     if (TOLOWE
1897     {
1898         main_s
1899         main_
1900         main_
1901 #ifdef FEAT_GU
1902         +initstr
1903 #endif
1904     }
1905
1906     if (TOLOWE
1907     {
1908         main_s
1909         main_
1910         main_
1911 #ifdef FEAT_GU
1912         +initstr
1913 #endif
1914     }
1915
1916     if (TOLOWE
1917     {
1918         main_s
1919         main_
1920         main_
1921 #ifdef FEAT_GU
1922         +initstr
1923 #endif
1924     }
1925
1926     if (TOLOWE
1927     {
1928         main_s
1929         main_
1930         main_
1931 #ifdef FEAT_GU
1932         +initstr
1933 #endif
1934     }
1935
1936     if (TOLOWE
1937     {
1938         main_s
1939         main_
1940         main_
1941 #ifdef FEAT_GU
1942         +initstr
1943 #endif
1944     }
1945
1946     if (TOLOWE
1947     {
1948         main_s
1949         main_
1950         main_
1951 #ifdef FEAT_GU
1952         +initstr
1953 #endif
1954     }
1955
1956     if (TOLOWE
1957     {
1958         main_s
1959         main_
1960         main_
1961 #ifdef FEAT_GU
1962         +initstr
1963 #endif
1964     }
1965
1966     if (TOLOWE
1967     {
1968         main_s
1969         main_
1970         main_
1971 #ifdef FEAT_GU
1972         +initstr
1973 #endif
1974     }
1975
1976     if (TOLOWE
1977     {
1978         main_s
1979         main_
1980         main_
1981 #ifdef FEAT_GU
1982         +initstr
1983 #endif
1984     }
1985
1986     if (TOLOWE
1987     {
1988         main_s
1989         main_
1990         main_
1991 #ifdef FEAT_GU
1992         +initstr
1993 #endif
1994     }
1995
1996     if (TOLOWE
1997     {
1998         main_s
1999         main_
2000         main_
2001 #ifdef FEAT_GU
2002         +initstr
2003 #endif
2004     }
2005
2006     if (TOLOWE
2007     {
2008         main_s
2009         main_
2010         main_
2011 #ifdef FEAT_GU
2012         +initstr
2013 #endif
2014     }
2015
2016     if (TOLOWE
2017     {
2018         main_s
2019         main_
2020         main_
2021 #ifdef FEAT_GU
2022         +initstr
2023 #endif
2024     }
2025
2026     if (TOLOWE
2027     {
2028         main_s
2029         main_
2030         main_
2031 #ifdef FEAT_GU
2032         +initstr
2033 #endif
2034     }
2035
2036     if (TOLOWE
2037     {
2038         main_s
2039         main_
2040         main_
2041 #ifdef FEAT_GU
2042         +initstr
2043 #endif
2044     }
2045
2046     if (TOLOWE
2047     {
2048         main_s
2049         main_
2050         main_
2051 #ifdef FEAT_GU
2052         +initstr
2053 #endif
2054     }
2055
2056     if (TOLOWE
2057     {
2058         main_s
2059         main_
2060         main_
2061 #ifdef FEAT_GU
2062         +initstr
2063 #endif
2064     }
2065
2066     if (TOLOWE
2067     {
2068         main_s
2069         main_
2070         main_
2071 #ifdef FEAT_GU
2072         +initstr
2073 #endif
2074     }
2075
2076     if (TOLOWE
2077     {
2078         main_s
2079         main_
2080         main_
2081 #ifdef FEAT_GU
2082         +initstr
2083 #endif
2084     }
2085
2086     if (TOLOWE
2087     {
2088         main_s
2089         main_
2090         main_
2091 #ifdef FEAT_GU
2092         +initstr
2093 #endif
2094     }
2095
2096     if (TOLOWE
2097     {
2098         main_s
2099         main_
2100         main_
2101 #ifdef FEAT_GU
2102         +initstr
2103 #endif
2104     }
2105
2106     if (TOLOWE
2107     {
2108         main_s
2109         main_
2110         main_
2111 #ifdef FEAT_GU
2112         +initstr
2113 #endif
2114     }
2115
2116     if (TOLOWE
2117     {
2118         main_s
2119         main_
2120         main_
2121 #ifdef FEAT_GU
2122         +initstr
2123 #endif
2124     }
2125
2126     if (TOLOWE
2127     {
2128         main_s
2129         main_
2130         main_
2131 #ifdef FEAT_GU
2132         +initstr
2133 #endif
2134     }
2135
2136     if (TOLOWE
2137     {
2138         main_s
2139         main_
2140         main_
2141 #ifdef FEAT_GU
2142         +initstr
2143 #endif
2144     }
2145
2146     if (TOLOWE
2147     {
2148         main_s
2149         main_
2150         main_
2151 #ifdef FEAT_GU
2152         +initstr
2153 #endif
2154     }
2155
2156     if (TOLOWE
2157     {
2158         main_s
2159         main_
2160         main_
2161 #ifdef FEAT_GU
2162         +initstr
2163 #endif
2164     }
2165
2166     if (TOLOWE
2167     {
2168         main_s
2169         main_
2170         main_
2171 #ifdef FEAT_GU
2172         +initstr
2173 #endif
2174     }
2175
2176     if (TOLOWE
2177     {
2178         main_s
2179         main_
2180         main_
2181 #ifdef FEAT_GU
2182         +initstr
2183 #endif
2184     }
2185
2186     if (TOLOWE
2187     {
2188         main_s
2189         main_
2190         main_
2191 #ifdef FEAT_GU
2192         +initstr
2193 #endif
2194     }
2195
2196     if (TOLOWE
2197     {
2198         main_s
2199         main_
2200         main_
2201 #ifdef FEAT_GU
2202         +initstr
2203 #endif
2204     }
2205
2206     if (TOLOWE
2207     {
2208         main_s
2209         main_
2210         main_
2211 #ifdef FEAT_GU
2212         +initstr
2213 #endif
2214     }
2215
2216     if (TOLOWE
2217     {
2218         main_s
2219         main_
2220         main_
2221 #ifdef FEAT_GU
2222         +initstr
2223 #endif
2224     }
2225
2226     if (TOLOWE
2227     {
2228         main_s
2229         main_
2230         main_
2231 #ifdef FEAT_GU
2232         +initstr
2233 #endif
2234     }
2235
2236     if (TOLOWE
2237     {
2238         main_s
2239         main_
2240         main_
2241 #ifdef FEAT_GU
2242         +initstr
2243 #endif
2244     }
2245
2246     if (TOLOWE
2247     {
2248         main_s
2249         main_
2250         main_
2251 #ifdef FEAT_GU
2252         +initstr
2253 #endif
2254     }
2255
2256     if (TOLOWE
2257     {
2258         main_s
2259         main_
2260         main_
2261 #ifdef FEAT_GU
2262         +initstr
2263 #endif
2264     }
2265
2266     if (TOLOWE
2267     {
2268         main_s
2269         main_
2270         main_
2271 #ifdef FEAT_GU
2272         +initstr
2273 #endif
2274     }
2275
2276     if (TOLOWE
2277     {
2278         main_s
2279         main_
2280         main_
2281 #ifdef FEAT_GU
2282         +initstr
2283 #endif
2284     }
2285
2286     if (TOLOWE
2287     {
2288         main_s
2289         main_
2290         main_
2291 #ifdef FEAT_GU
2292         +initstr
2293 #endif
2294     }
2295
2296     if (TOLOWE
2297     {
2298         main_s
2299         main_
2300         main_
2301 #ifdef FEAT_GU
2302         +initstr
2303 #endif
2304     }
2305
2306     if (TOLOWE
2307     {
2308         main_s
2309         main_
2310         main_
2311 #ifdef FEAT_GU
2312         +initstr
2313 #endif
2314     }
2315
2316     if (TOLOWE
2317     {
2318         main_s
2319         main_
2320         main_
2321 #ifdef FEAT_GU
2322         +initstr
2323 #endif
2324     }
2325
2326     if (TOLOWE
2327     {
2328         main_s
2329         main_
2330         main_
2331 #ifdef FEAT_GU
2332         +initstr
2333 #endif
2334     }
2335
2336     if (TOLOWE
2337     {
2338         main_s
2339         main_
2340         main_
2341 #ifdef FEAT_GU
2342         +initstr
2343 #endif
2344     }
2345
2346     if (TOLOWE
2347     {
2348         main_s
2349         main_
2350         main_
2351 #ifdef FEAT_GU
2352         +initstr
2353 #endif
2354     }
2355
2356     if (TOLOWE
2357     {
2358         main_s
2359         main_
2360         main_
2361 #ifdef FEAT_GU
2362         +initstr
2363 #endif
2364     }
2365
2366     if (TOLOWE
2367     {
2368         main_s
2369         main_
2370         main_
2371 #ifdef FEAT_GU
2372         +initstr
2373 #endif
2374     }
2375
2376     if (TOLOWE
2377     {
2378         main_s
2379         main_
2380         main_
2381 #ifdef FEAT_GU
2382         +initstr
2383 #endif
2384     }
2385
2386     if (TOLOWE
2387     {
2388         main_s
2389         main_
2390         main_
2391 #ifdef FEAT_GU
2392         +initstr
2393 #endif
2394     }
2395
2396     if (TOLOWE
2397     {
2398         main_s
2399         main_
2400         main_
2401 #ifdef FEAT_GU
2402         +initstr
2403 #endif
2404     }
2405
2406     if (TOLOWE
2407     {
2408         main_s
2409         main_
2410         main_
2411 #ifdef FEAT_GU
2412         +initstr
2413 #endif
2414     }
2415
2416     if (TOLOWE
2417     {
2418         main_s
2419         main_
2420         main_
2421 #ifdef FEAT_GU
2422         +initstr
2423 #endif
2424     }
2425
2426     if (TOLOWE
2427     {
2428         main_s
2429         main_
2430         main_
2431 #ifdef FEAT_GU
2432         +initstr
2433 #endif
2434     }
2435
2436     if (TOLOWE
2437     {
2438         main_s
2439         main_
2440         main_
2441 #ifdef FEAT_GU
2442         +initstr
2443 #endif
2444     }
2445
2446     if (TOLOWE
2447     {
2448         main_s
2449         main_
2450         main_
2451 #ifdef FEAT_GU
2452         +initstr
2453 #endif
2454     }
2455
2456     if (TOLOWE
2457     {
2458         main_s
2459         main_
2460         main_
2461 #ifdef FEAT_GU
2462         +initstr
2463 #endif
2464     }
2465
2466     if (TOLOWE
2467     {
2468         main_s
2469         main_
2470         main_
2471 #ifdef FEAT_GU
2472         +initstr
2473 #endif
2474     }
2475
2476     if (TOLOWE
2477     {
2478         main_s
2479         main_
2480         main_
2481 #ifdef FEAT_GU
2482         +initstr
2483 #endif
2484     }
2485
2486     if (TOLOWE
2487     {
2488         main_s
2489         main_
2490         main_
2491 #ifdef FEAT_GU
2492         +initstr
2493 #endif
2494     }
2495
2496     if (TOLOWE
2497     {
2498         main_s
2499         main_
2500         main_
2501 #ifdef FEAT_GU
2502         +initstr
2503 #endif
2504     }
2505
2506     if (TOLOWE
2507     {
2508         main_s
2509         main_
2510         main_
2511 #ifdef FEAT_GU
2512         +initstr
2513 #endif
2514     }
2515
2516     if (TOLOWE
2517     {
2518         main_s
2519         main_
2520         main_
2521 #ifdef FEAT_GU
2522         +initstr
2523 #endif
2524     }
2525
2526     if (TOLOWE
2527     {
2528         main_s
2529         main_
2530         main_
2531 #ifdef FEAT_GU
2532         +initstr
2533 #endif
2534     }
2535
2536     if (TOLOWE
2537     {
2538         main_s
2539         main_
2540         main_
2541 #ifdef FEAT_GU
2542         +initstr
2543 #endif
2544     }
2545
2546     if (TOLOWE
2547     {
2548         main_s
2549         main_
2550         main_
2551 #ifdef FEAT_GU
2552         +initstr
2553 #endif
2554     }
2555
2556     if (TOLOWE
2557     {
2558         main_s
2559         main_
2560         main_
2561 #ifdef FEAT_GU
2562         +initstr
2563 #endif
2564     }
2565
2566     if (TOLOWE
2567     {
2568         main_s
2569         main_
2570         main_
2571 #ifdef FEAT_GU
2572         +initstr
2573 #endif
2574     }
2575
2576     if (TOLOWE
2577     {
2578         main_s
2579         main_
2580         main_
2581 #ifdef FEAT_GU
2582         +initstr
2583 #endif
2584     }
2585
2586     if (TOLOWE
2587     {
2588         main_s
2589         main_
2590         main_
2591 #ifdef FEAT_GU
2592         +initstr
2593 #endif
2594     }
2595
2596     if (TOLOWE
2597     {
2598         main_s
2599         main_
2600         main_
2601 #ifdef FEAT_GU
2602         +initstr
2603 #endif
2604     }
2605
2606     if (TOLOWE
2607     {
2608         main_s
2609         main_
2610         main_
2611 #ifdef FEAT_GU
2612         +initstr
2613 #endif
2614     }
2615
2616     if (TOLOWE
2617     {
2618         main_s
2619         main_
2620         main_
2621 #ifdef FEAT_GU
2622         +initstr
2623 #endif
2624     }
2625
2626     if (TOLOWE
2627     {
2628         main_s
2629         main_
2630         main_
2631 #ifdef FEAT_GU
2632         +initstr
2633 #endif
2634     }
2635
2636     if (TOLOWE
2637     {
2638         main_s
2639         main_
2640         main_
2641 #ifdef FEAT_GU
2642         +initstr
2643 #endif
2644     }
2645
2646     if (TOLOWE
2647     {
2648         main_s
2649         main_
2650         main_
2651 #ifdef FEAT_GU
2652         +initstr
2653 #endif
2654     }
2655
2656     if (TOLOWE
2657     {
2658         main_s
2659         main_
2660         main_
2661 #ifdef FEAT_GU
2662         +initstr
2663 #endif
2664     }
2665
2666     if (TOLOWE
2667     {
2668         main_s
2669         main_
2670         main_
2671 #ifdef FEAT_GU
2672         +initstr
2673 #endif
2674     }
2675
2676     if (TOLOWE
2677     {
2678         main_s
2679         main_
2680         main_
2681 #ifdef FEAT_GU
2682         +initstr
2683 #endif
2684     }
2685
2686     if (TOLOWE
2687     {
2688         main_s
2689         main_
2690         main_
2691 #ifdef FEAT_GU
2692         +initstr
2693 #endif
2694     }
2695
2696     if (TOLOWE
2697     {
2698         main_s
2699         main_
2700         main_
2701 #ifdef FEAT_GU
2702         +initstr
2703 #endif
2704     }
2705
2706     if (TOLOWE
2707     {
2708         main_s
2709         main_
2710         main_
2711 #ifdef FEAT_GU
2712         +initstr
2713 #endif
2714     }
2715
2716     if (TOLOWE
2717     {
2718         main_s
2719         main_
2720         main_
2721 #ifdef FEAT_GU
2722         +initstr
2723 #endif
2724     }
2725
2726     if (TOLOWE
2727     {
2728         main_s
2729         main_
2730         main_
2731 #ifdef FEAT_GU
2732         +initstr
2733 #endif
2734     }
2735
2736     if (TOLOWE
2737     {
2738         main_s
2739         main_
2740         main_
2741 #ifdef FEAT_GU
2742         +initstr
2743 #endif
2744     }
2745
2746     if (TOLOWE
2747     {
2748         main_s
2749         main_
2750         main_
2751 #ifdef FEAT_GU
2752         +initstr
2753 #endif
2754     }
2755
2756     if (TOLOWE
2757     {
2758         main_s
2759         main_
2760         main_
2761 #ifdef FEAT_GU
2762         +initstr
2763 #endif
2764     }
2765
2766     if (TOLOWE
2767     {
2768         main_s
2769         main_
2770         main_
2771 #ifdef FEAT_GU
2772         +initstr
2773 #endif
2774     }
2775
2776     if (TOLOWE
2777     {
2778         main_s
2779         main_
2780         main_
2781 #ifdef FEAT_GU
2782         +initstr
2783 #endif
2784     }
2785
2786     if (TOLOWE
2787     {
2788         main_s
2789         main_
2790         main_
2791 #ifdef FEAT_GU
2792         +initstr
2793 #endif
2794     }
2795
2796     if (TOLOWE
2797     {
2798         main_s
2799         main_
2800         main_
2801 #ifdef FEAT_GU
2802         +initstr
2803 #endif
2804     }
2805
2806     if (TOLOWE
2807     {
2808         main_s
2809         main_
2810         main_
2811 #ifdef FEAT_GU
2812         +initstr
2813 #endif
2814     }
2815
2816     if (TOLOWE
2817     {
2818         main_s
2819         main_
2820         main_
2821 #ifdef FEAT_GU
2822         +initstr
2823 #endif
2824     }
2825
2826     if (TOLOWE
2827     {
2828         main_s
2829         main_
2830         main_
2831 #ifdef FEAT_GU
2832         +initstr
2833 #endif
2834     }
2835
2836     if (TOLOWE
2837     {
2838         main_s
2839         main_
2840         main_
2841 #ifdef FEAT_GU
2842         +initstr
2843 #endif
2844     }
2845
2846     if (TOLOWE
2847     {
2848         main_s
2849         main_
2850         main_
2851 #ifdef FEAT_GU
2852         +initstr
2853 #endif
2854     }
2855
2856     if (TOLOWE
2857     {
2858         main_s
2859         main_
2860         main_
2861 #ifdef FEAT_GU
2862         +initstr
2863 #endif
2864     }
2865
2866     if (TOLOWE
2867     {
2868         main_s
2869         main_
2870         main_
2871 #ifdef FEAT_GU
2872         +initstr
2873 #endif
2874     }
2875
2876     if (TOLOWE
2877     {
2878         main_s
2879         main_
2880         main_
2881 #ifdef FEAT_GU
2882         +initstr
2883 #endif
2884     }
2885
2886     if (TOLOWE
2887     {
2888         main_s
2889         main_
2890         main_
2891 #ifdef FEAT_GU
2892         +initstr
2893 #endif
2894     }
2895
2896     if (TOLOWE
2897     {
2898         main_s
2899         main_
2900         main_
2901 #ifdef FEAT_GU
2902         +initstr
2903 #endif
2904     }
2905
2906     if (TOLOWE
2907     {
2908         main_s
2909         main_
2910         main_
2911 #ifdef FEAT_GU
2912         +initstr
2913 #endif
2914     }
2915
2916     if (TOLOWE
2917     {
2918         main_s
2919         main_
2920         main_
2921 #ifdef FEAT_GU
2922         +initstr
2923 #endif
2924     }
2925
2926     if (TOLOWE
2927     {
2928         main_s
2929         main_
2930         main_
2931 #ifdef FEAT_GU
2932         +initstr
2933 #endif
2934     }
2935
2936     if (TOLOWE
2937     {
2938         main_s
2939         main_
2940         main_
2941 #ifdef FEAT_GU
2942         +initstr
2943 #endif
2944     }
2945
2946     if (TOLOWE
2947     {
2948         main_s
2949         main_
2950         main_
2951 #ifdef FEAT_GU
2952         +initstr
2953 #endif
2954     }
2955
2956     if (TOLOWE
2957     {
2958         main_s
2959         main_
2960         main_
2961 #ifdef FEAT_GU
2962         +initstr

```

教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题

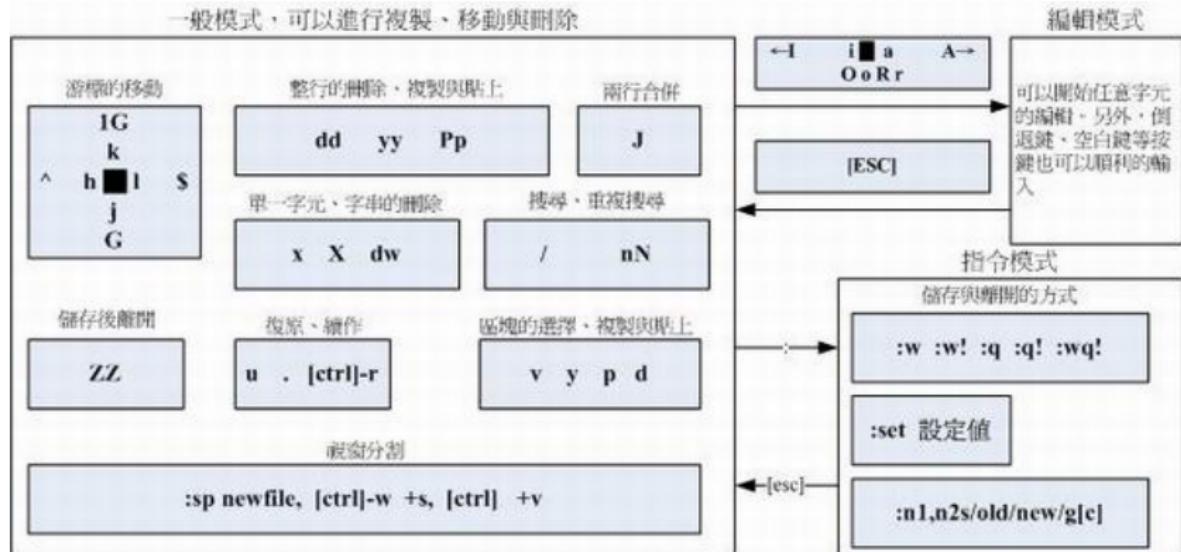


Vim 命令集锦 | 命令组合

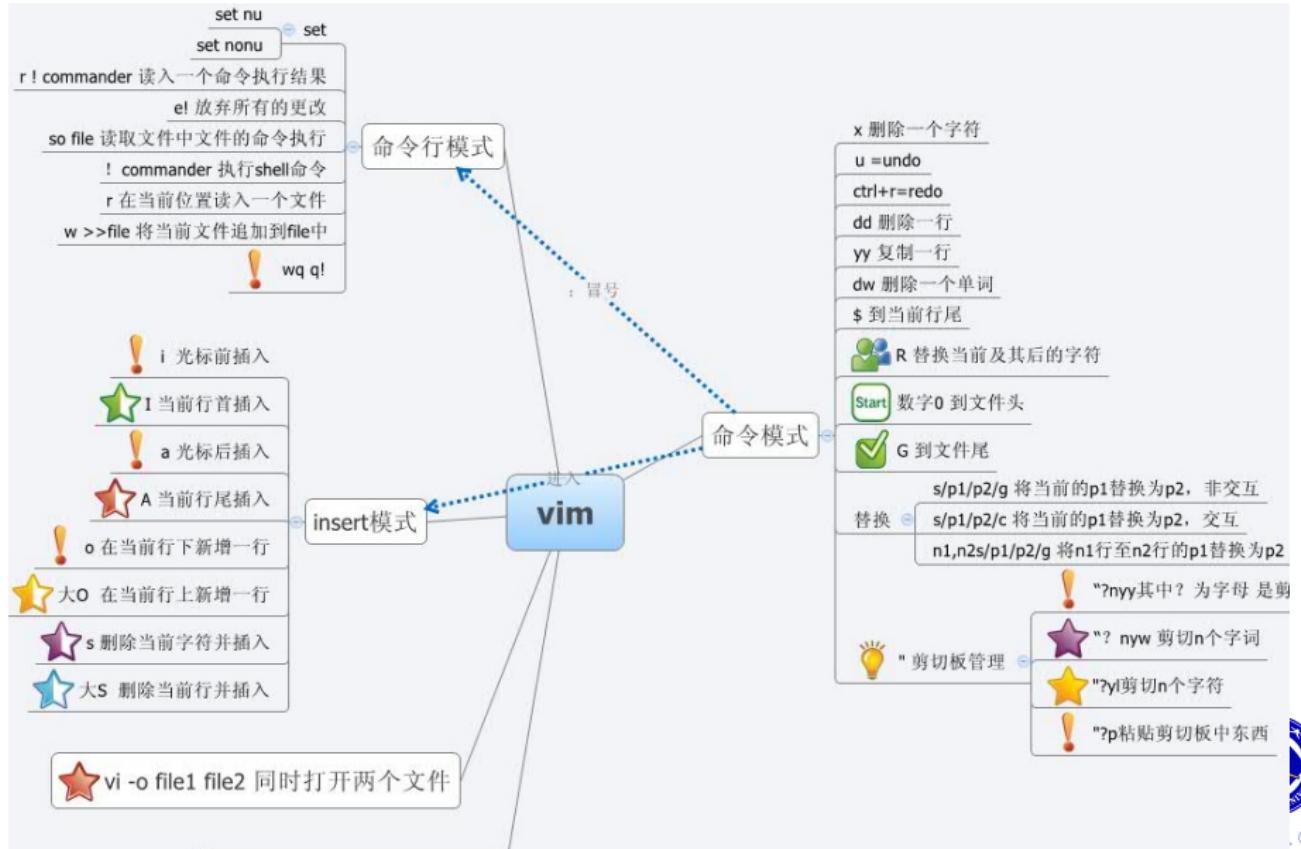
P aste	O pen	Change	delete	yank	*
u ndo	o pen	cc	dd	yy	
single line		c3↑	d5↓	yz↑	
multi-lines		cw	dw	yw	
word					/usr/share/dict/linux.words



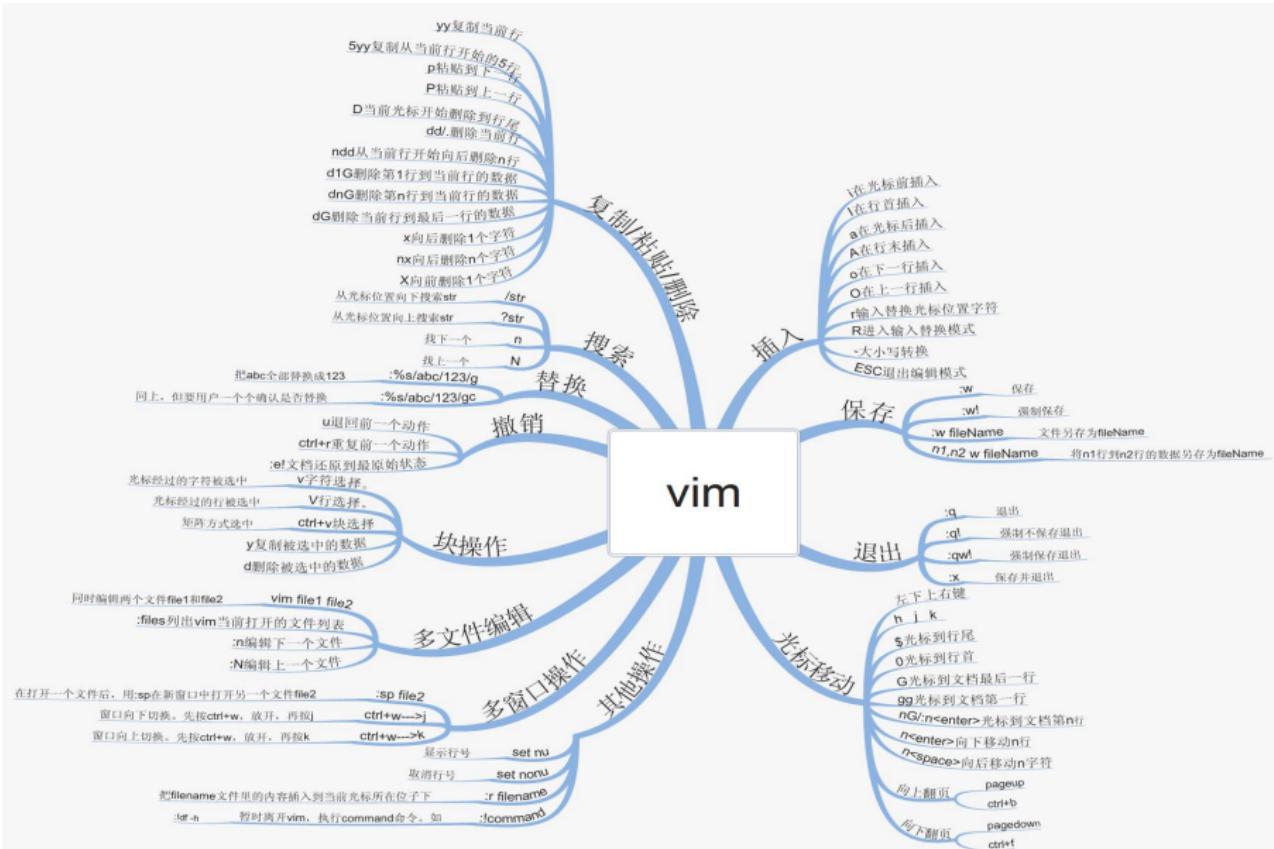
Vim 命令集錦



Vim 命令集锦



Vim 命令集锦



Vim 命令集锦

```
private boolean foo;
private boolean #_ReallyLongName;
private boolean aReallyShortName;
private int bar1, bar2;
private boolean isFooAndBar(){
    foo = false;
}
public void main(String args[]){
    foo = true;
    if(foo){
        bar1 = bar2 + 1 + fooClass.invokeRandomMethod();
        bar1 = bar2 + 2;
    }
    bar1++;
    bar2++;
    if( aReallyLongName
        aReallyLongName
        aReallyShortName
    )
}
```

Buffers:

- Sample.java [+]
- AnotherNew.java
- AnotherNew.java 3,0-1
- /fooCl

Registers:

- H zt
- Ctrl-B
- M zz
- Ctrl-F
- L zb

Visual Mode:

- h ← j ↑ k ↓ l →
- w ← b ← Ctrl-N
- gd
- Ctrl-W p
- Ctrl-W j
- Ctrl-W k
- Ctrl-W l
- vsplit
- diffsplit

File Status:

- 23, 29
- 83%

Bottom Status:

- Created by vgod, Dec. 2009

Vim 命令集锦



vim - movement commands

Vim 命令集锦

~ toggle case	! external filter	@ play macro	# prev ident	\$ eol	% goto match	^ "soft" bol	& repeat :s	* next ident	(begin sentence) end sentence	"soft" bol down	+ next line
~. goto mark	1 2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0	2 3 4 5 6 7 8 9 0
Q ex mode	W next WORD	E end WORD	R replace mode	T 'till	Y yank line	U undo line	I insert at bol	O open above	P paste before	{ begin parag.	} end parag.	auto format
Q record macro	W next word	E end word	R replace char	T 'till	Y yank	U undo	I insert mode	O open below	P paste after	{ misc	} misc	
A append at col	S subst line	D deleted to col	F "back" find ch	G eof/ goto ln	H screen top	J join lines	K help	L screen bottom	: ex cmd line	!! reg. spec	bol/ goto col	
a append	S subst char	d delete	f find char	g extra cmds	h ←	j ↓	k ↑	l →	: repeat ; t/T/f/F	! goto mk. bol	\ not used!	
Z quit ^	X back-space	C change to col	V visual lines	B prev WORD	N prev (find)	M screen mid'	< un indent	> indent	? find (rev.)			
Z extra ^	X delete char	C change ^	V visual mode	b prev word	n next (find)	m set mark	, reverse , t/T/f/F	. repeat cmd	/ find			

基本	h j k l
w e b	按word移动
行内	fc 搜索下一个c tc一样 光标在左 0 ^ \$ 跳到一行的头 开始尾
移动	Ctrl-f 向下移动一屏 Ctrl-b 向上移动一屏 G 到文件尾 gg 到文件首 12G 到指定的(12)行
全文	H/M/L 光标到屏幕上/中/下 zt 光标处滚到屏幕上部 zz 光标处滚到屏幕中部 zb 光标处滚到屏幕底部
搜索	* / # 搜索光标处的字符串 /word 向后搜索word ?word 向前搜索word n 重复搜索 . 到上次编辑文件的地方

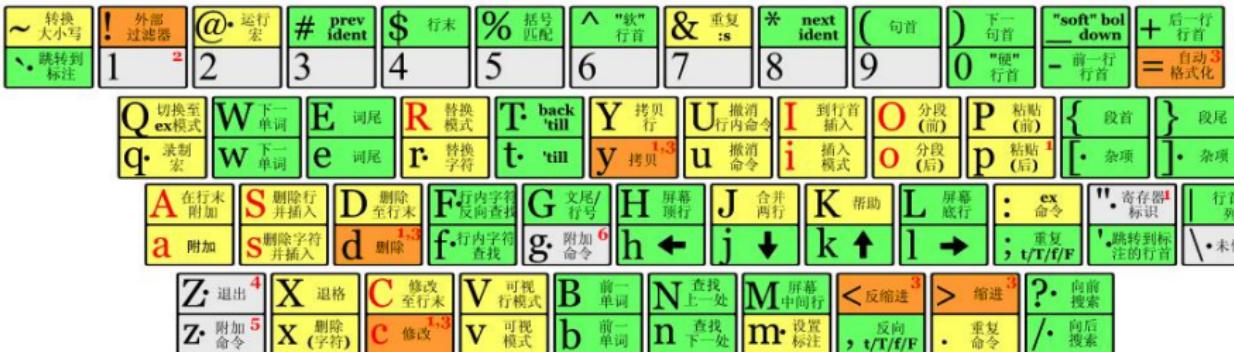
编辑	d{motion} 剪切 c{motion} 剪切插入 y{motion} 复制 P 粘贴
	dd 剪切当前行 cc 剪切 插入模式 yy / Y 复制当前行
	D/C 剪切从光标位置到行尾 x 剪切字符 s剪切插入
	r 替换当前字符 R 连续替换 .
代码	Ctrl-n 自动补全 ~ 大小写切换 >> 缩进所有选择的代码
	<< 反缩进 = 自动缩进 jp 自动调整粘贴 % 匹配花括号、方括号 gd 到达光标处函数变量的定义处

Vim 命令集锦

vi / vim 键盘图

Esc

命令模式



动作

移动光标, 或者定义操作的范围

命令

直接执行的命令,
红色命令 进入编辑模式

操作

后面跟随表示操作范围的指令

extra

特殊功能,
需要额外的输入

q·

后跟字符参数

w,e,b命令

小写(b): `:guux([foo], bar, baz);`
大写(B): `:guux(foo, bar, baz);`

主要ex命令:

:w (保存), :q (退出), :q! (不保存退出)
:e f (打开文件 f),
:%s/x/y/g ('y' 全局替换 'x'),
:h (帮助 in vim), :new (新建文件 in vim)

其它重要命令:

CTRL-R: 重复 (vim),
CTRL-F/B: 上翻/下翻,
CTRL-E/-Y: 上滚/下滚,
CTRL-V: 块可视模式 (vim only)

可视模式:

漫游后对选中的区域执行操作 (vim only)

备注:

- (1) 在拷贝/粘贴/删除 命令前使用 "x (x=a..z,")
使用命令的寄存器('剪贴板')
(如: "ay\$ 拷贝剩余的行内容至寄存器 'a')
- (2) 命令前添加数字
多遍重复操作
(e.g.: 2p, d2w, 5i, d4j)
- (3) 重复本字符在光标所在行执行操作
(dd = 删除本行, >> = 行首缩进)
- (4) ZZ 保存退出, ZQ 不保存退出
- (5) zt: 移动光标所在行至屏幕顶端,
zb: 底端, zz: 中间
- (6) gg: 文首 (vim only),
gf: 打开光标处的文件名 (vim only)

Vim 命令集锦 | 键盘布局



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

- 回顾与总结
- 总结
- 思考题



动词

动词代表了我们打算对文本进行什么样的操作。

名词

名词代表了我们即将处理的文本对象 (text object) 。

介词

介词界定了待编辑文本的范围或者位置。



动词

动词代表了我们打算对文本进行什么样的操作。

名词

名词代表了我们即将处理的文本对象 (text object) 。

介词

介词界定了待编辑文本的范围或者位置。



动词

动词代表了我们打算对文本进行什么样的操作。

名词

名词代表了我们即将处理的文本对象 (text object) 。

介词

介词界定了待编辑文本的范围或者位置。



动词

- d : delete, 删除
- r : replace, 替换
- c : change, 修改
- y : yank, 复制
- v : visual select, 选取

名词

- w : word, 单词
- s : sentence, 句子
- p : paragraph, 段落
- t : tag, HTML 标签

介词

- i : inside, 在…之内
- a : around, 环绕…
- t : to, 到…位置前
- f : forward, 到…位置上



动词

- d : delete, 删除
- r : replace, 替换
- c : change, 修改
- y : yank, 复制
- v : visual select, 选取

名词

- w : word, 单词
- s : sentence, 句子
- p : paragraph, 段落
- t : tag, HTML 标签

介词

- i : inside, 在…之内
- a : around, 环绕…
- t : to, 到…位置前
- f : forward, 到…位置上



动词

- d : delete, 删除
- r : replace, 替换
- c : change, 修改
- y : yank, 复制
- v : visual select, 选取

名词

- w : word, 单词
- s : sentence, 句子
- p : paragraph, 段落
- t : tag, HTML 标签

介词

- i : inside, 在…之内
- a : around, 环绕…
- t : to, 到…位置前
- f : forward, 到…位置上



基本语法

动词 + 介词 + 名词

实例

- dip : delete inside paragraph, 删除一个段落
- vis : visual select inside sentence, 选区一个句子
- ciw : change inside word, 修改一个单词
- caw : change around word, 修改一个单词
- dtx : delete to x, 删除文本直到字符 “x” 前 (不包括字符 “x”)
- dfx : delete forward x, 删除文本直到字符 “x” 上 (包括字符 “x”)



基本语法

动词 + 介词 + 名词

实例

- dip : delete inside paragraph, 删除一个段落
- vis : visual select inside sentence, 选区一个句子
- ciw : change inside word, 修改一个单词
- caw : change around word, 修改一个单词
- dtx : delete to x, 删除文本直到字符 “x” 前 (不包括字符 “x”)
- dfx : delete forward x, 删除文本直到字符 “x” 上 (包括字符 “x”)



数词

数词指定了待编辑文本对象的数量，从这个角度而言，数词也可以看作是一种介词。

造句语法

动词 + 介词/数词 + 名词

实例

- c3w : change three words, 修改三个单词
- d2w : delete two words, 删除两个单词



数词

数词指定了待编辑文本对象的数量，从这个角度而言，数词也可以看作是一种介词。

造句语法

动词 + 介词/数词 + 名词

实例

- c3w : change three words, 修改三个单词
- d2w : delete two words, 删除两个单词



数词

数词指定了待编辑文本对象的数量，从这个角度而言，数词也可以看作是一种介词。

造句语法

动词 + 介词/数词 + 名词

实例

- c3w : change three words, 修改三个单词
- d2w : delete two words, 删除两个单词



补充

数词也可以修饰动词，表示将操作执行 n 次。

对应语法

数词 + 动词 + 名词

实例

- 2dw : twice delete word, 两次删除单词（等价于删除两个单词）
- 3x : three times delete character, 三次删除字符（等价于删除三个字符）



补充

数词也可以修饰动词，表示将操作执行 n 次。

对应语法

数词 + 动词 + 名词

实例

- 2dw : twice delete word, 两次删除单词（等价于删除两个单词）
- 3x : three times delete character, 三次删除字符（等价于删除三个字符）



补充

数词也可以修饰动词，表示将操作执行 n 次。

对应语法

数词 + 动词 + 名词

实例

- 2dw : twice delete word, 两次删除单词（等价于删除两个单词）
- 3x : three times delete character, 三次删除字符（等价于删除三个字符）



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



知识点

- Vim 的主要工作模式：命令模式、输入模式、末行模式
- Vim 的启动和退出：启动，保存，退出
- Vim 中的移动和定位
- Vim 中的文本编辑：进入输入模式，修改删除、复制粘贴、搜索替换
- 在 Vim 中运行系统命令
- Vim 的学习：用户手册，帮助系统，命令总结

技能

- 使用 Vim 进行日常的文本编辑



教学提纲

1 引言

2 Vim 简介

- vi 版本
- 学习 Vim
- 启动 Vim
- 状态行
- 工作模式

3 移动和定位

4 编辑文件

- 进入输入模式
- 剪切删除
- 修改替换

- 复制粘贴
- 撤销重做
- 搜索替换
- 保存退出
- 命令补遗

5 Vim 进阶

- 运行命令
- Vim 插件

6 Vim 命令集锦

7 Vim 语言

8 回顾与总结

- 总结
- 思考题



- ① Vim 的工作模式主要有哪三种？
- ② 在 Vim 中如何进行模式的转换？
- ③ Vim 中最基本的移动命令是哪四个？
- ④ 进入 Vim 输入模式的命令有哪些？
- ⑤ 在 Vim 中如何进行剪切、复制和粘贴？
- ⑥ 在 Vim 中如何进行撤销和重做？
- ⑦ 在 Vim 中如何进行搜索和替换？
- ⑧ 在 Vim 中如何运行系统命令？
- ⑨ 启动、保存文件和退出 Vim 的命令有哪些？



下节预告

还记得学习过的 C 语言吗？试着编写几个小程序，读取输入、条件判断、循环迭代、编写函数、……



Powered by



T_EX L^AT_EX X_ET_EX Beamer