

# 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

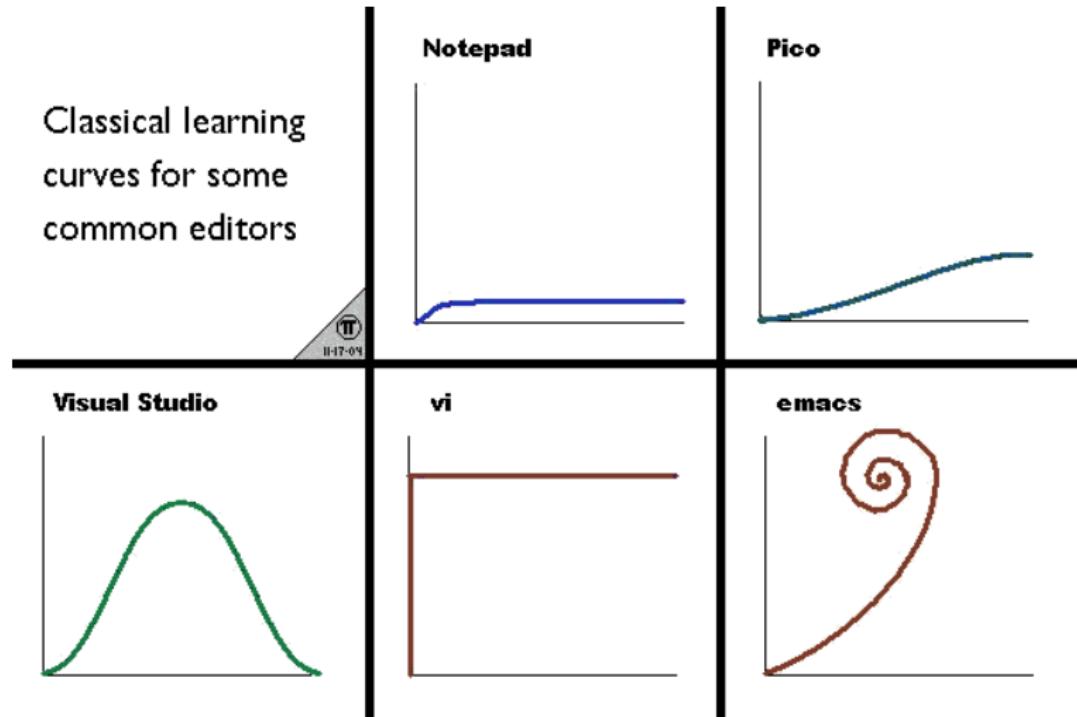
## 7 Vim 语言

- 回顾与总结
- 总结
- 思考题



## 引言 | 纯文本 vs. 格式化文本





## 引言 | 文本编辑器

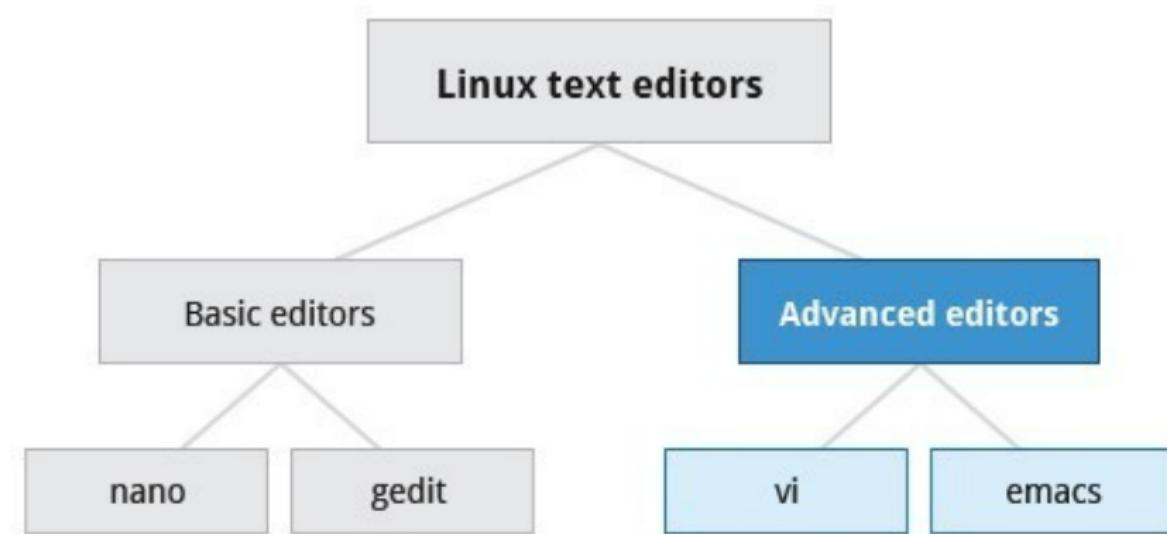
```
[No file] - GVIM
File Edit Tools Syntax Buffers Window
Help

VIM - Vi IMproved
version 6.0.152
by Bram Moolenaar et al.
Vim is open source and freely distributable

      Help poor children in Uganda!
type :help iccc<Enter>      for information

type :q<Enter>              to exit
type :help<Enter> or <F1>   for on-line help
type :help version6<Enter>   for version info
0..0-1 All
```





## 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

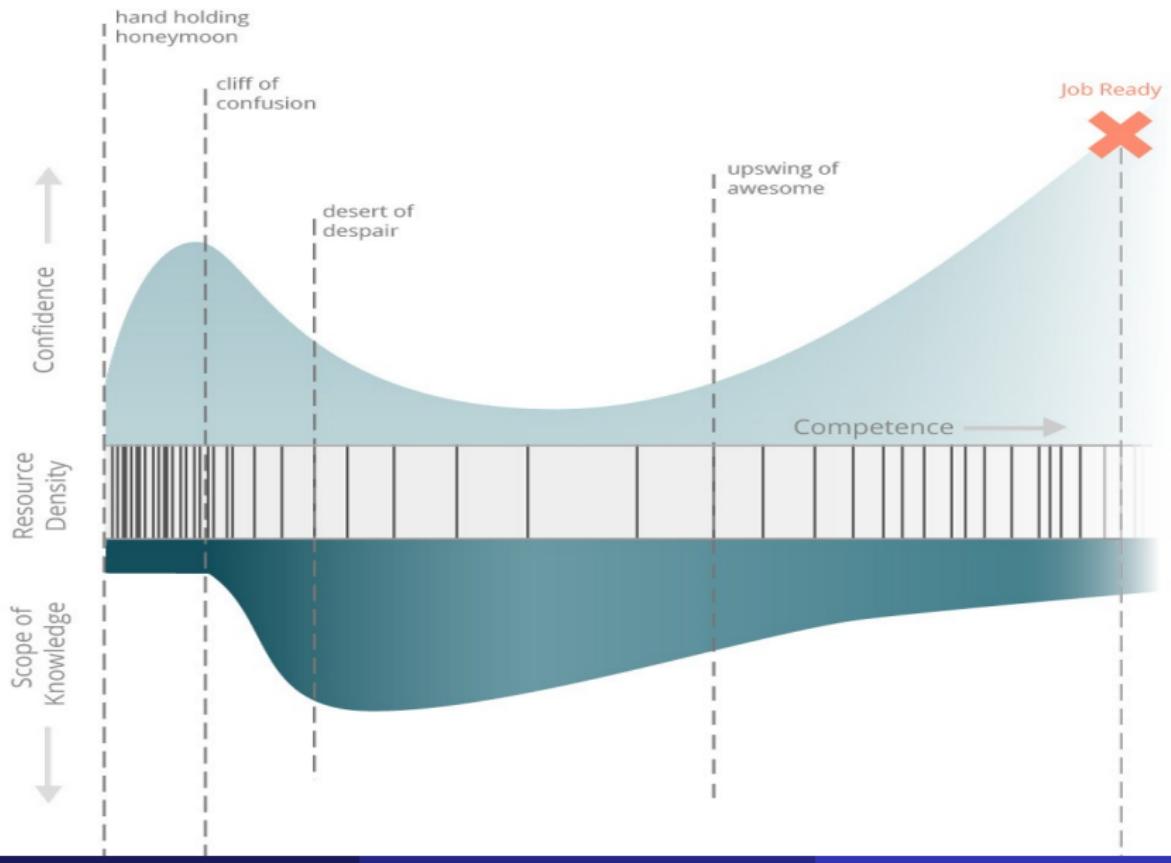
## 大腕版文本编辑器

“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

## 8 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：vimtutor

Vim 用户手册 在 Vim 中输入：`:help user-manual@cn`（中文版），  
`:help user-manual`（英文版）

Vim 帮助系统 在 Vim 中输入：`:help`

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`”来阅读主要帮助文件。



## 入门学习

Vim 初学者教学 在 Linux 系统命令行下输入：vimtutor

Vim 用户手册 在 Vim 中输入：`:help user-manual@cn`（中文版），  
`:help user-manual`（英文版）

Vim 帮助系统 在 Vim 中输入：`:help`

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。



## 入门学习

Vim 初学者教学 在 Linux 系统命令行下输入：vimtutor

Vim 用户手册 在 Vim 中输入：`:help user-manual@cn`（中文版），  
`:help user-manual`（英文版）

Vim 帮助系统 在 Vim 中输入：`:help`

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 status items:

- 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

8 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

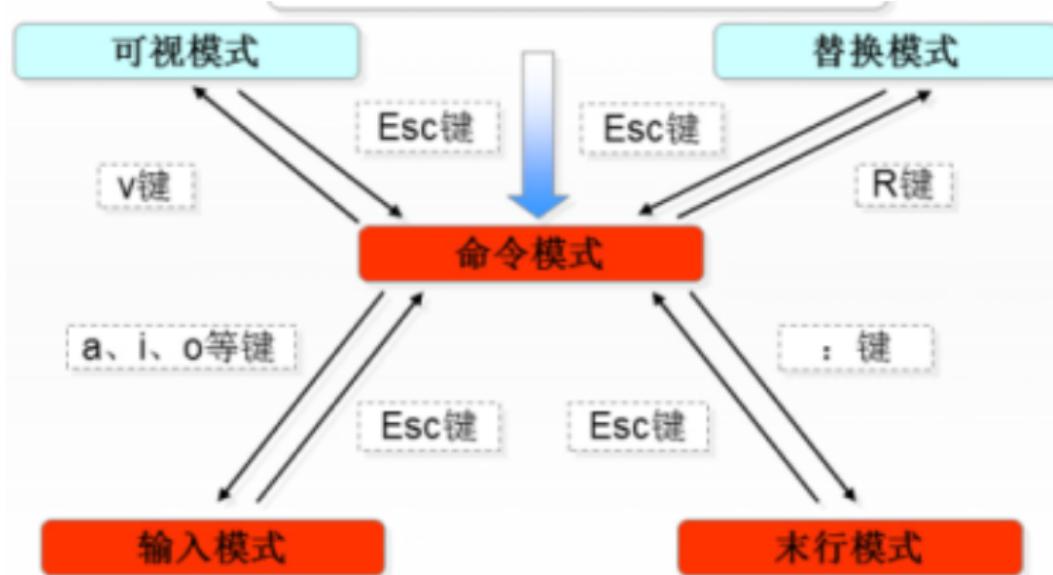
启动vi编辑器后默认进入命令模式，该模式下主要完成如光标移动、字符串查找、删除、复制、粘贴等操作。不论用户处于何种模式，只要按下Esc键，即可进入命令模式。

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

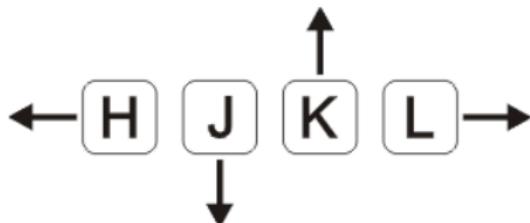
- 回顾与总结
- 总结
- 思考题



# 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 (:\$)
xG	G 前加数字	光标移至指定行
:x	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



# 编辑 | 剪切删除

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

- 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

- 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-1] [-]  
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%]  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
  
2 [Line:1/1,Column:1] [100%]  
3 [Line:1/1,Column:1] [100%]  
4 [Line:1/1,Column:1] [100%]  
5 [Line:1/1,Column:1] [100%]  
6 [Line:1/1,Column:1] [100%]

# 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

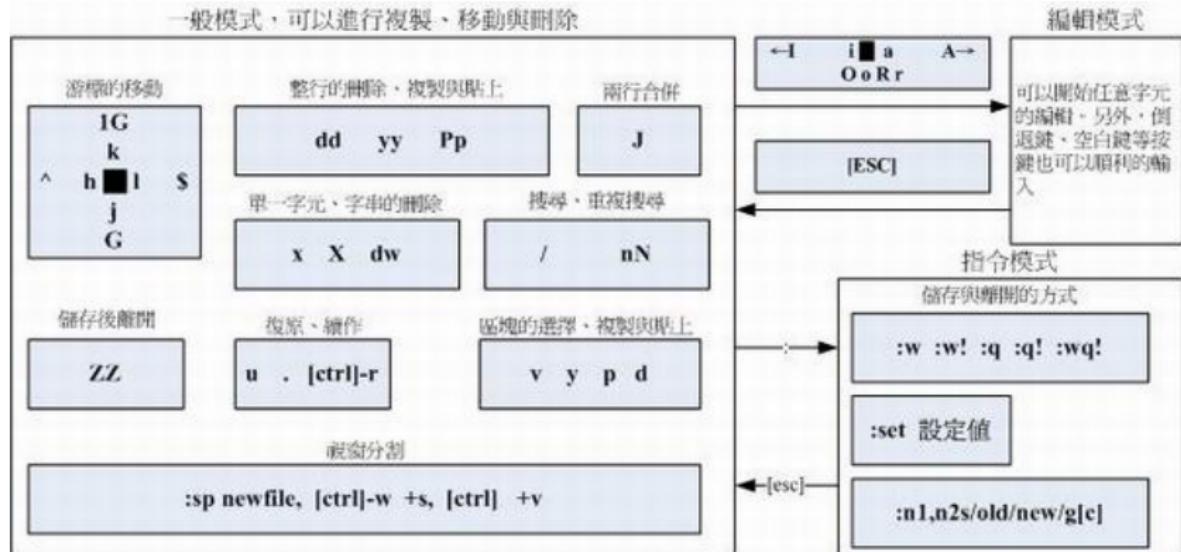


# Vim 命令集锦 | 命令组合

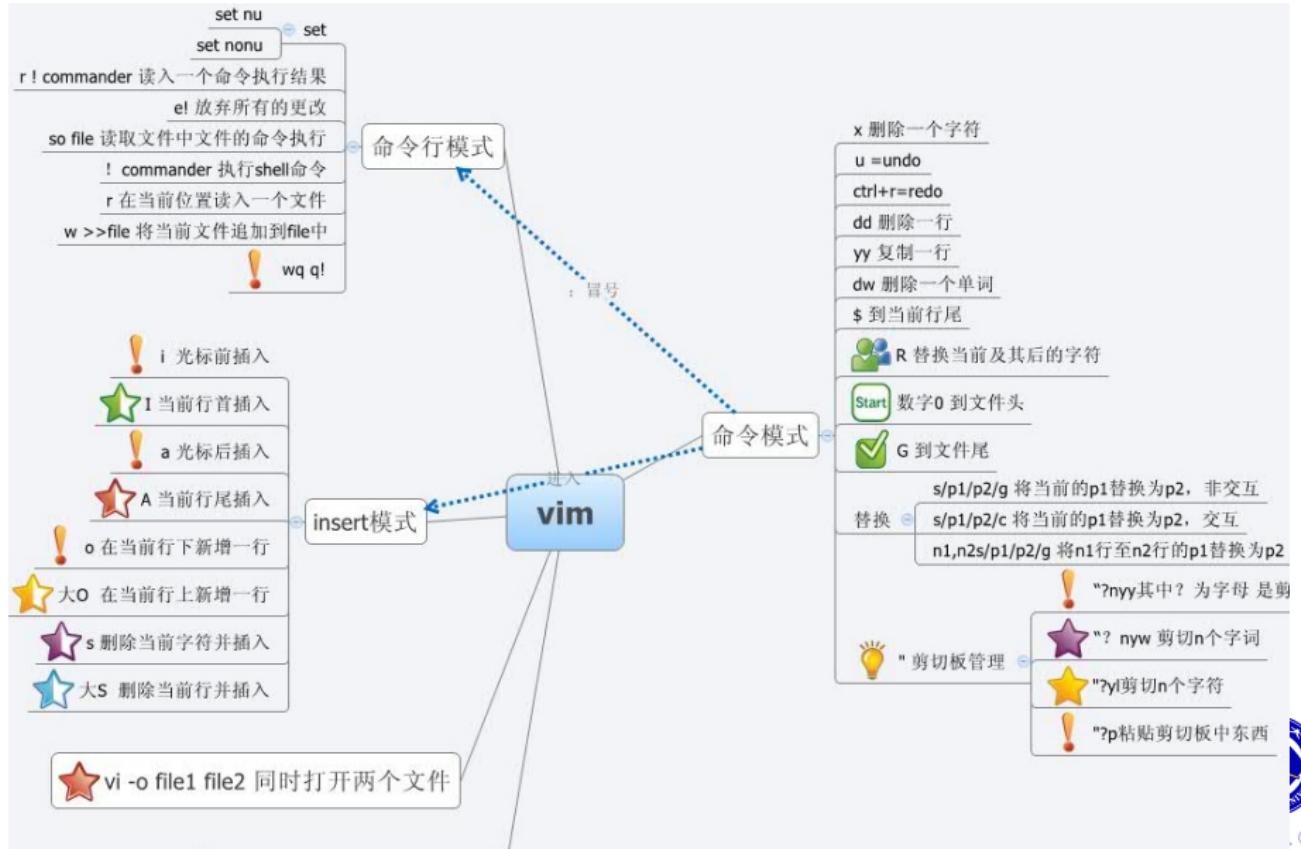
P aste	O pen	Change	delete	yank	*
u ndo	o pen	cc	dd	yy	
single line		c3↑	d5↓	yz↑	
multi-lines		cw	dw	yw	
word					/usr/share/dict/linux.words



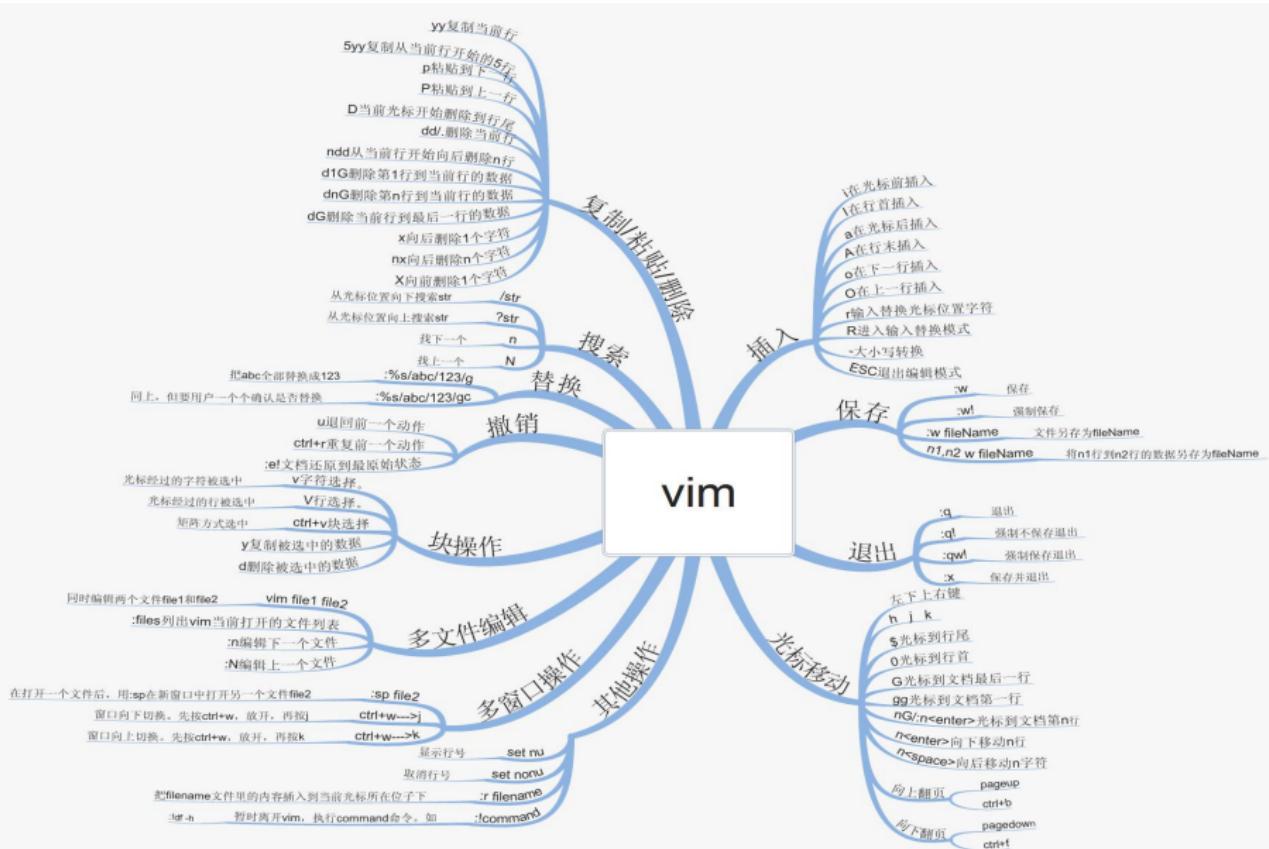
# Vim 命令集錦



# Vim 命令集锦



# Vim 命令集锦



# Vim 命令集锦

The diagram illustrates Vim command flow across multiple windows and buffers. Key features shown include:

- Buffer Navigation:** Commands like `zt`, `zg`, `zG`, `zv`, `zV`, `zB`, `zF`, `zL`, and `zB` are used to switch between buffers.
- Text Selection and Movement:** Commands like `h`, `j`, `k`, `l`, `w`, `b`, `Ctrl-N`, `gd`, `mx`, `Ctrl-W p`, `Ctrl-W j`, `Ctrl-W k`, `L`, and `zg` are used for navigating through text.
- File Operations:** Commands like `split`, `vsplit`, `diffsplit`, and `top` are used for managing windows and buffers.
- Search and Replace:** Commands like `Ctrl-F` and `Ctrl-H` are used for search and replace operations.
- Visual Mode:** Commands like `Ctrl-V` and `Ctrl-A` are used in visual mode.

The code snippets shown in the buffers include Java code with annotations for specific Vim commands being demonstrated.

Buffers and their content:

- `Sample.java`:  
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;  
 if(  
 aReallyLongName  
 aReallyLongName  
 aReallyShortName  
 )  
 mx  
 }  
}
- `AnotherNew.java`:  
import java.awt.\*;  
import java.awt.event.\*;  
public class AnotherNew  
{  
 /  
 /fooCl  
 /fooCl  
}
- `Top Another.java`:  
import java.awt.\*;  
import java.awt.event.\*;  
public class Another  
{  
 /  
 /fooCl  
 /fooCl  
}

Bottom status bar details:

- AnotherNew.java 3,0-1 Top Another.java 3,0-1
- Created by vgod, Dec. 2009
- Yxf (TJMU)
- Vim
- 2018年6月 70/87

## 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	not used!
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		
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 cmd	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

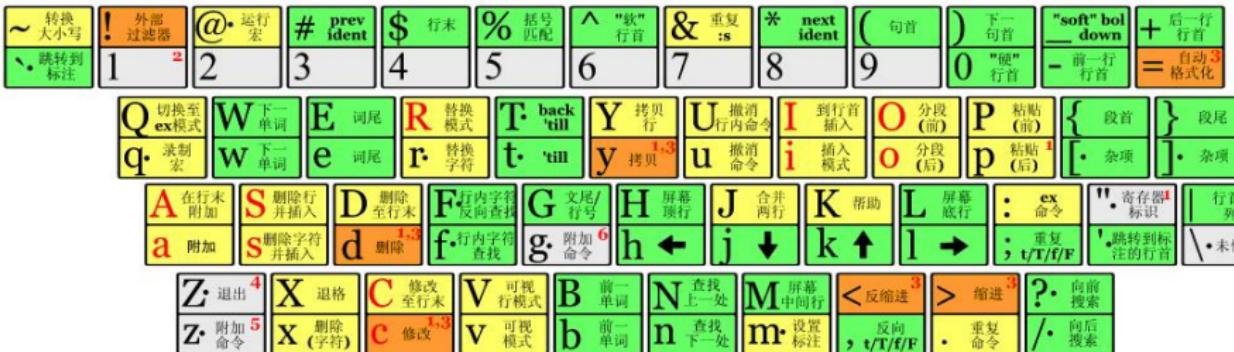
编辑	d{motion} 剪切 c{motion} 剪切插入 y{motion} 复制 P 粘贴
	dd 剪切当前行 cc 剪切 插入模式 yy / Y 复制当前行
	D/C 剪切从光标位置到行尾 x 剪切字符 s剪切插入
	r 替换当前字符 R 连续替换 .
代码	Ctrl-n 自动补全 ~ 大小写切换 >> 缩进所有选择的代码
	<< 反缩进 = 自动缩进 jp 自动调整粘贴
	% 匹配花括号、方括号 gd 到达光标处函数变量的定义处

# 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: "`a$y` 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) 重复本字符串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<sub>E</sub>X L<sup>A</sup>T<sub>E</sub>X X<sub>E</sub>T<sub>E</sub>X Beamer