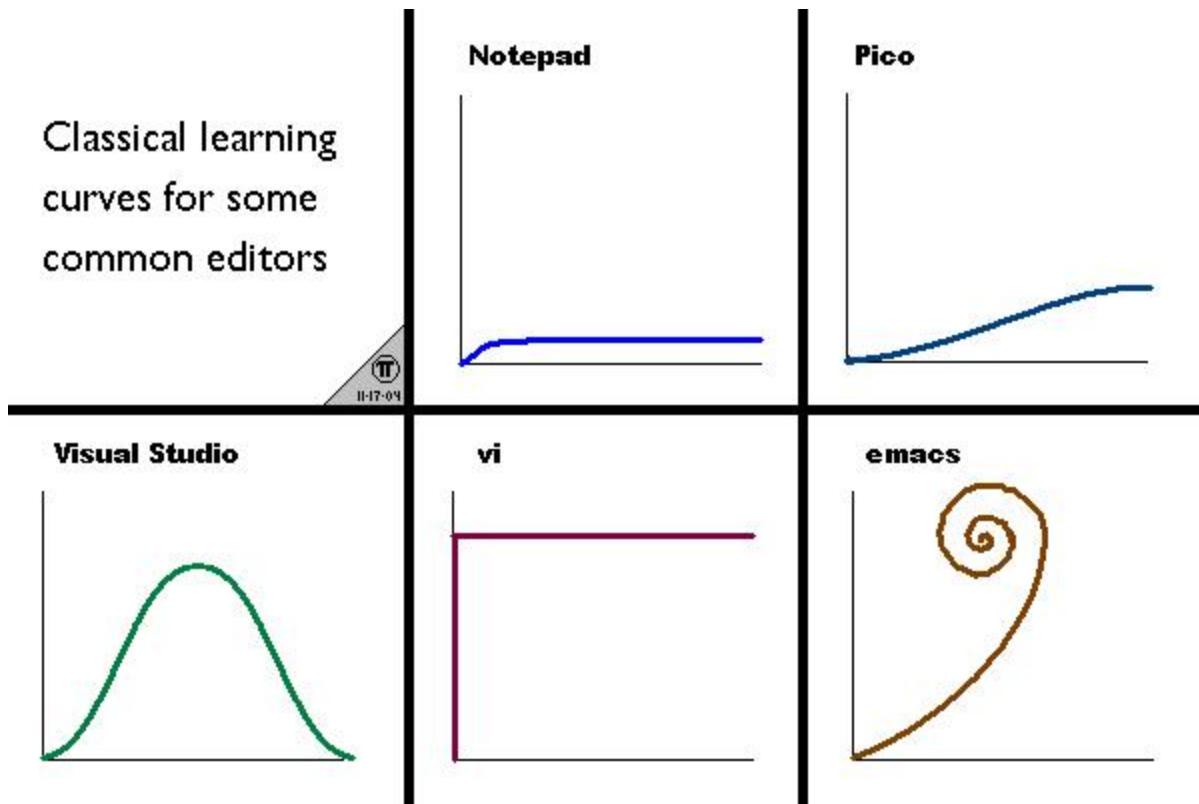


关于vim

-- by [@wklken](#)

学习曲线



编辑器

文本编辑器, 不是IDE

能做一些事情, 但是一些事情是做不到的, 不要强求, 该用IDE的时候, 用就是了

入门步骤

from 简明 练级攻略

存活

感觉良好

觉得更好，更强，更快

使用VIM的超能力

过程

一个vimer必定会经历的过程

1. 什么都没有, 纯vi
2. 什么都有
3. 只留适合自己的, 不适合自己也要配置成适合自己的
4. 什么都没有(听说)

诀窍

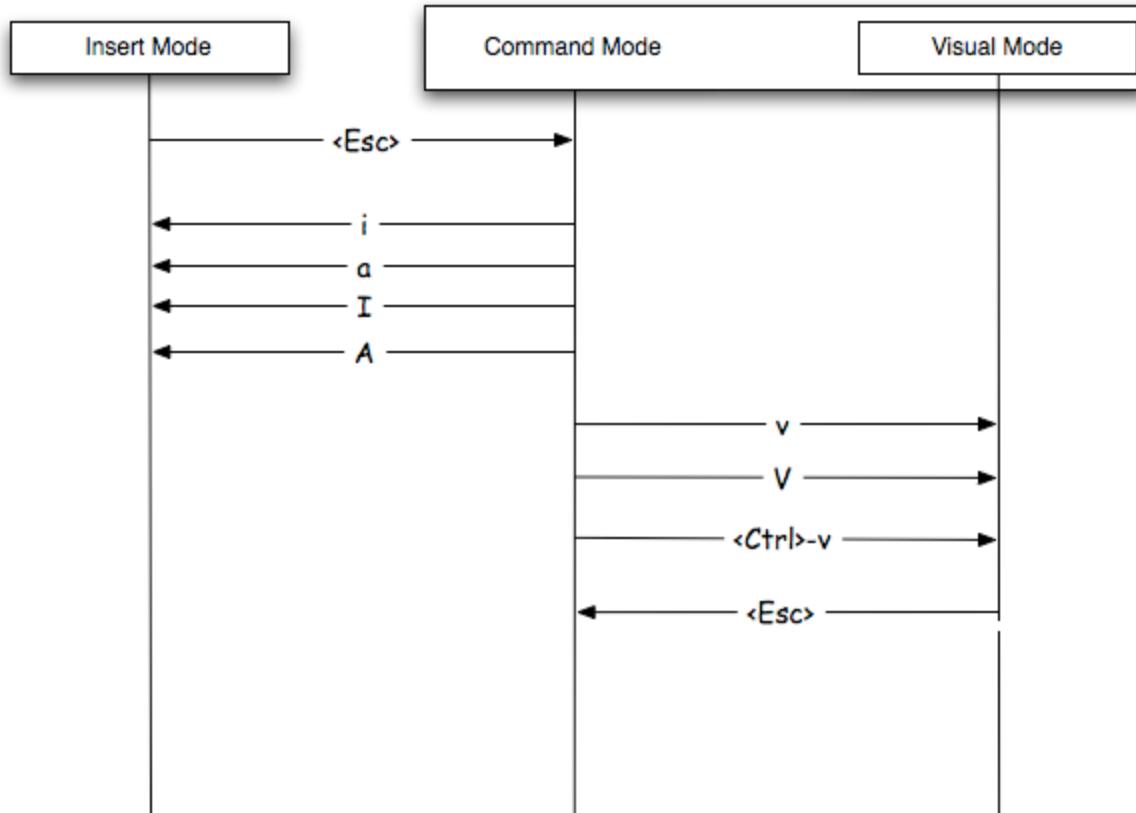
不断练习

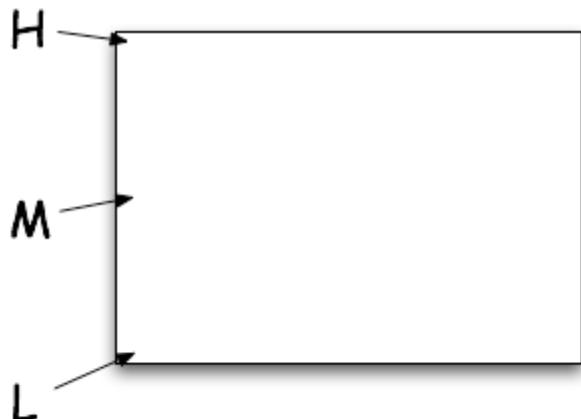
- 肌肉记忆!
- 直觉 -> 行动, 而不是 思考 -> 行动(十倍差距)

目的

- 实操演示: vim能做什么
- 思考: 哪些是你常用编辑器可以实现的? 哪些好的功能是你想要却没有的? 你的痛点是什么? 如何实现一个更为高效的工作流?
- 二八原则: 一些常见实用的操作

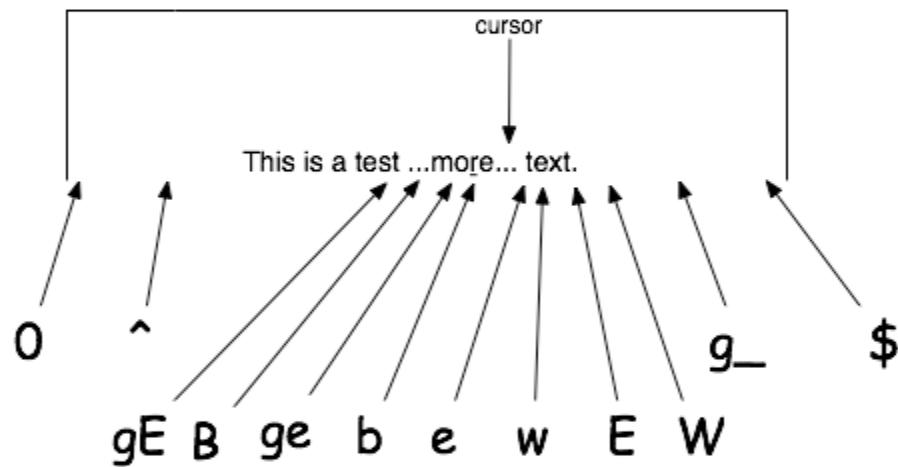
vim入门 - 模式



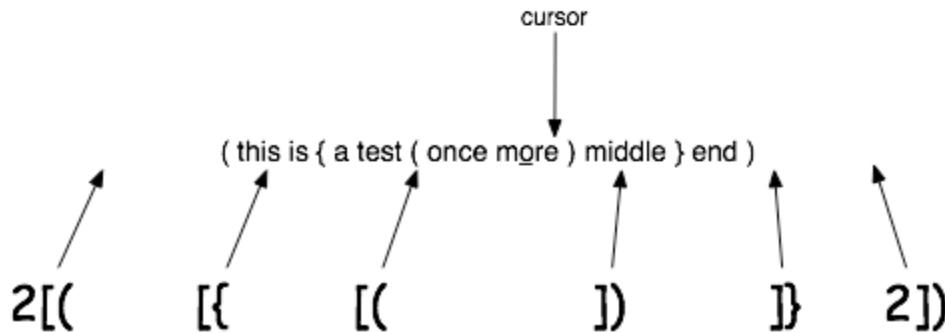


↑ Ctrl-e (1 line)
 Ctrl-d (1/2 page)
 Ctrl-f (1 page)

↓ Ctrl-y (1 line)
 Ctrl-u (1/2 page)
 Ctrl-b (1 page)



g0, g^ and g\$ work for screen-lines (if the line is longer than the screen).



These commands work over many lines! Use % to jump to the matching bracket.

vim入门 - 移动

忘掉刚才那张图.....

vim 入门 - 移动

- hjkl (请杜绝方向键, 移动右手到方向键区浪费时间)

```
map <Left> <N>
map <Right> <N>
map <Up> <N>
map <Down> <N>
```

- 单词: w / b / e (忘记: W/B/E-以空白为分隔符, 要多按shift键键/不好记/用得少)
- 行内: 0 / \$ (忘记: ^, 够不着啊)
- 段⁺⁺: { / }

vim入门 - 移动

- 页: <ctrl-u> / <ctrl-d> (忘记: ctrl-f/ctrl-b, 都在左侧键区左手太累, 经常是要上要下, 而不是往一个方向走)
- 可视范围: H M L (全部忘记, 没多大用, 要按shift, 混淆记忆)
- 文件内: gg / G / :N (基本够了)
- 匹配处: % 括号匹配 / # (忘记: *, 在键盘左侧, 太远不好敲, 可以交换#和*的功能)

更强大的 => 看后面的插件部分

vim入门 - 编辑

| N<action>

- 删除字符: x
- 删除整行: dd
- 删除: dw / db / d\$ / dG / dgg
- 复制: yw / yb / y\$ / yy
- 粘贴: p / P
- 合并: J

vim入门 - 撤销和重复

- 撤销: u
- 重复: .

vim入门 - 选中

- 选中: v => vw / vb / vta / v\$
- 块选中: V + hjkl
- 行首加东西 / 行尾加东西

vim入门 - 搜索

- 行内: f /t (可以选择性遗忘t, 同时, 忘记F/T, 甚至可以忘记f, 你不需要)
- 全局: :/ e d

vim入门 - 替换

记住这三个似乎够了(频率最高)

- :1,10s/a/b/g
- :%s/a/b/g
- # 选中, 或者 +范围 选中, :% //b/

字符替换

- r

vim入门 - 文本对象

初学者基本不了解的一个特性, 很多用了好多年也不知道的特性

<action>a<object> or <action>i<object>

- a = all / i = in
- action: d y v c
- object: (d) / (e e ce) / a a a
- object: " ']
- object: a
- object: 装插件可扩展 (e) / e(e e f e) / (de)

vim入门 - 分屏

- :sp file1
- :vsp file2
- ctrl-w-h/j/k/l (改键 ctrl-h/j/k/l)
- ctrl-w-H/J/K/L

vim 插件 - 语法检查

syntastic

vim插件 - 自动补全/代码高亮

效率神器!

- [YCM](#) : 毫秒级补全/ python / c系等, 编译安装, 具体自行文档
- [ultisnips](#) + [vim-snippets](#)
- [delimitMate](#) 括号补全
- [closetag](#) xml/html标签补全

vim插件 - 快速编码

- [nerdcommenter](#) 快速注释
- [vim-surround](#) + [vim-repeat](#) 快速编辑
- [vim-trailing-whitespace](#) 去空格
- [vim-easy-align](#) 代码对齐

vim插件 - 快速运行

- vim-quickrun

vim插件 - 快速移动

- vim-easymotion 移动神器
- vim-signature 标签

vim插件 - 快速选中

- [vim-expand-region](#) 区块大小
- [vim-multiple-cursors](#) 多光标选中编辑

vim插件 - 文件导航和搜索

- `ctrlspace` buffer导航
- `nerdtree` 目录导航
- `tagbar` 标签导航
- `ctrlp` 文件搜索
- `ctrlsf` 类sublimetext搜索编辑

关于插件

- 不是越多越好
- 配了用不上 = 没配 + 浪费资源
- 同一功能, 对比几个插件, 选择一个合适的
- 快捷键配置一定要容易记
- 相信我, 一个插件用的最多的快捷键就两个, 绝大多数情况下不会多于两个, 不用耗费心力在配置更强大的操作上
- 二八原则

关于插件

- 更符合自觉的键位/操作
- 尽量减少敲击次数
- 杜绝一切无效的敲击
- 个性化, 定制到每个细节, 力争解决自己所有痛点

怎么配置

- 参考别人的, 筛选需要的点, 读对应文档, 修改得到自己的
- 根据自己的痛点, 寻求解决方案, 最终个性化自己的配置

资源

配置:

- k-vim
- spf13
- maximum-awsome

插件库及主题库

- vimawesome
- vimcolors

Q & A

Thanks!