



# VIM入门与进阶

潘魏增@美团网

世界上的程序员有三种：一种用vim，一种用emacs，剩余的是其它。

# 关于我

- 了解接触vim 8年

# 关于我

- 了解接触vim 8年
- 完全用vim工作 3年

# 关于我

- 了解接触vim 8年
- 完全用vim工作 3年
- editplus  
dreamweaver  
visual studio  
eclipse  
vim

- `vim.prototype = new vi();`  
vim是vi文本编辑器的扩展版本
- 1976年大神Bill Joy开发了vi，给程序员世界带来了火种
- 1991年大牛Bram Moolenaar在vi基础上开发了vim

- vim.prototype = new vi(  
vim是vi文本编辑器的扩
- 1976年大神Bill Joy开发  
界带来了火种
- 1991年大牛Bram Moolenaar  
发了vim



- `vim.prototype = new vi();`  
vim是vi文本编辑器的扩展版本
- 1976年大神Bill Joy开发了vi，给程序员世界带来了火种
- 1991年大牛Bram Moolenaar在vi基础上开发了vim



```
= new vi();
```

编辑器的扩展版本

Bill Joy开发了vi，给程序员世

ram Moolenaar在vi基础上开

- `vim.prototype = new vi();`  
vim是vi文本编辑器的扩展版本
- 1976年大神Bill Joy开发了vi，给程序员世界带来了火种
- 1991年大牛Bram Moolenaar在vi基础上开发了vim

最大的优点

最大的优点

速度

# 其他优点

- 强大的定制性

# 其他优点

- 强大的定制性
- 性感到让人欲罢不能的配色

# 其他优点

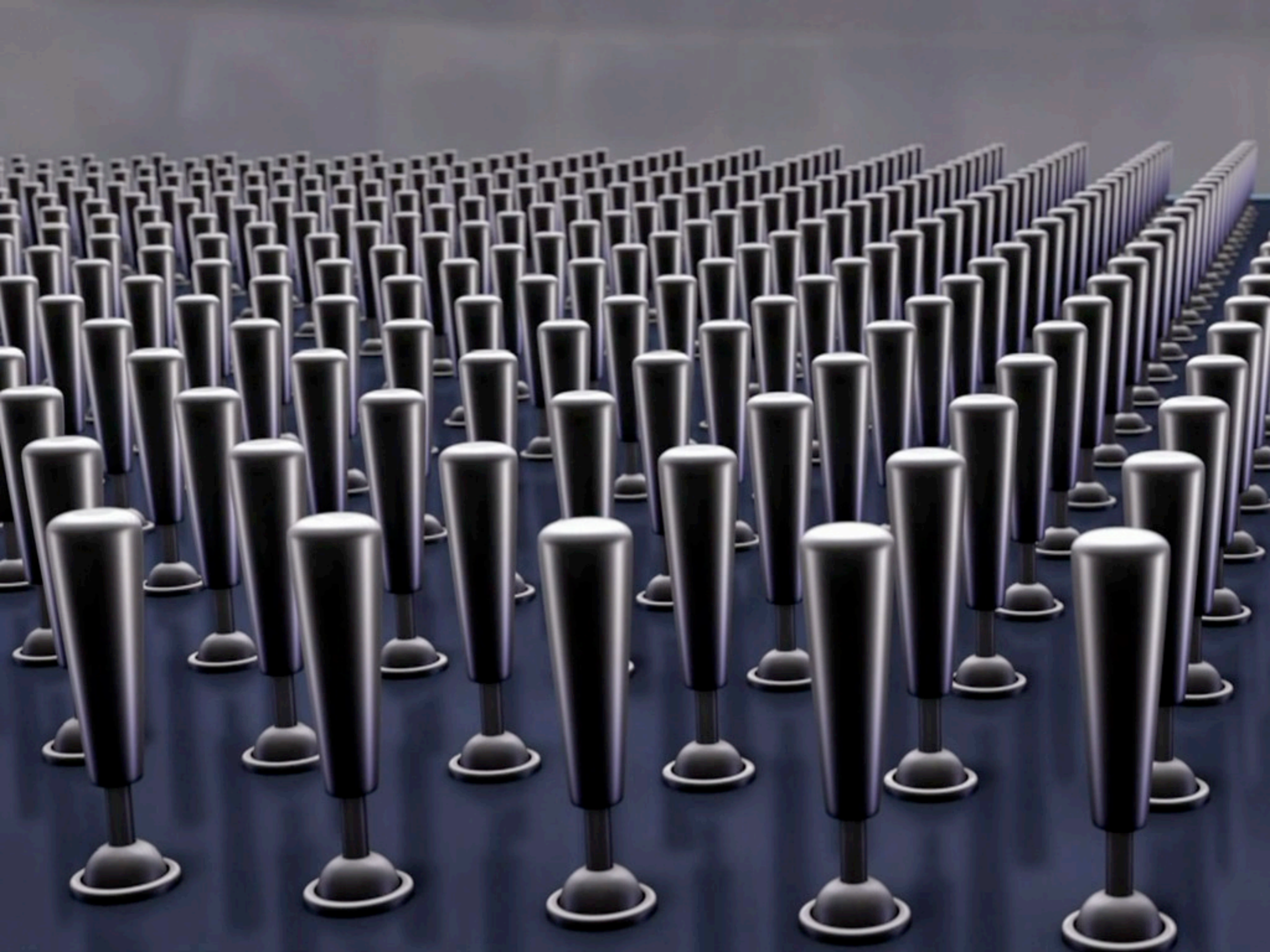
- 强大的定制性
- 性感到让人欲罢不能的配色
- 无限精彩 触手可及

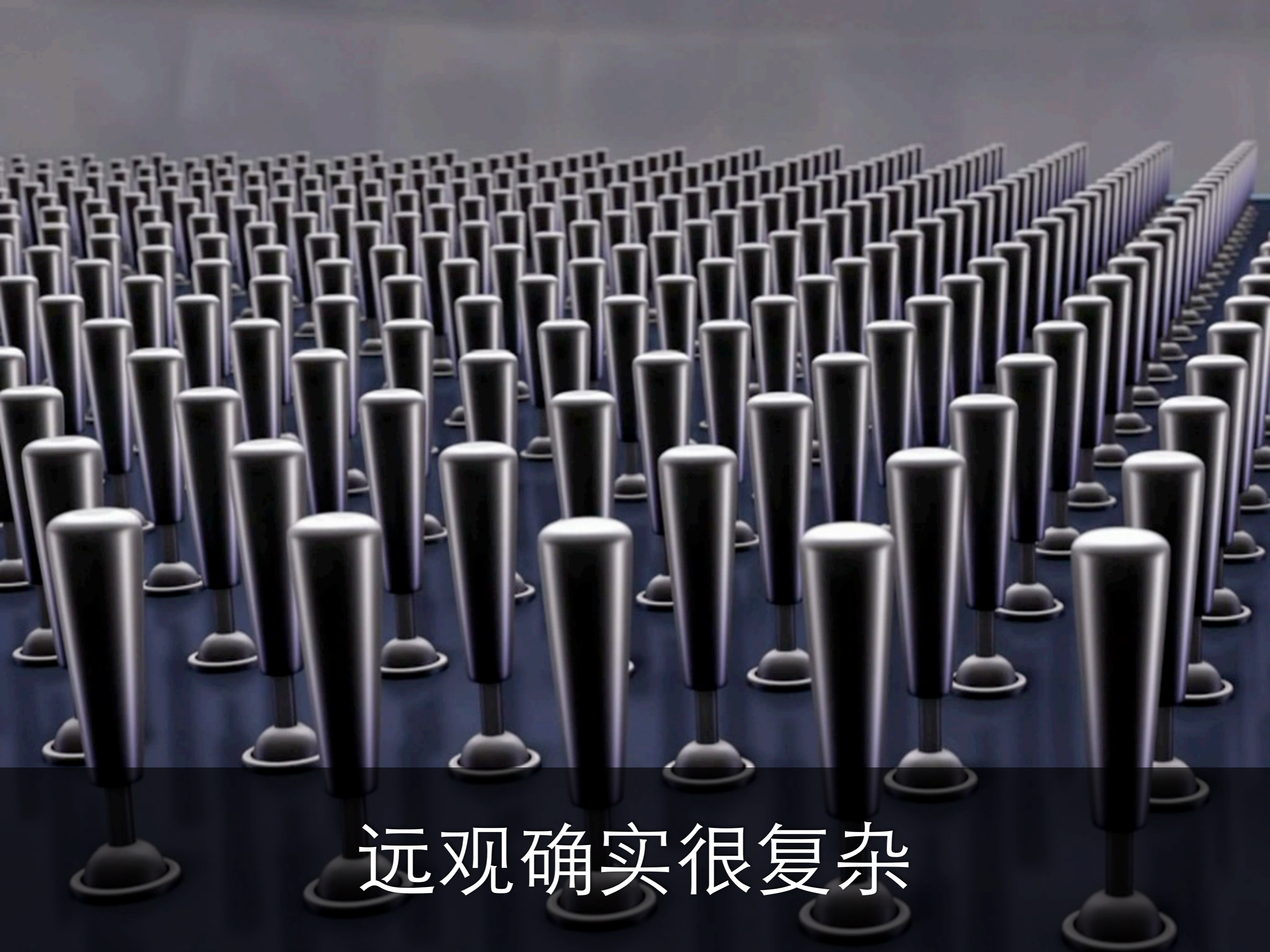
# vi / vim graphical cheat sheet

Esc

normal mode

~ 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 <sup>2</sup>	2	3	4	5	6	7	8	9	0 "hard" bol	- prev line	= auto <sup>3</sup> format
Q ex mode	W next WORD	E end WORD	R replace mode	T back 'till	Y yank line	U undo line	I insert at bol	O open above	P paste before	{ begin parag.	} end parag.	
q record macro	w next word	e end word	r replace char	t 'till	y yank <sup>1,3</sup>	u undo	i insert mode	o open below	p paste after <sup>1</sup>	[ misc	] misc	
A append at eol	S subst line	D delete to eol	F "back" find ch	G eof/ goto ln	H screen top	J join lines	K help	L screen bottom	. ex cmd line	" reg. spec <sup>1</sup>	bol/ goto col	
a append	s subst char	d delete <sup>1,3</sup>	f find char	g extra <sup>6</sup> cmds	h ←	j ↓	k ↑	l →	. repeat ; t/T/f/F	' goto mk. bol	\ not used!	
Z quit <sup>4</sup>	X back-space	C change to eol	V visual lines	B prev WORD	N prev (find)	M screen mid'l	< un-indent <sup>3</sup>	> indent <sup>3</sup>	? find (rev.)			
Z extra <sup>5</sup> cmds	X delete char	c change <sup>1,3</sup>	v visual mode	b prev word	n next (find)	m set mark	, reverse t/T/f/F	. repeat cmd	/ find			





远观确实很复杂

# 模式

- normal 普通模式
- insert 插入模式
- command 命令行模式
- visual 可视模式

# 基本命令

- `vi a.txt`
- `vimdiff a.txt b.txt`

# 基本命令

- `:w, :x, :w !sudo tee %`
- `:q, :qa, :q!`
- `:e a.txt, tabe a.txt`  
`:sp a.txt, :vsp a.txt`
- `:Sex`
- `:r!date, :r!git diff`

# 移动

- h, j, k, l
- ^, \$, %, f, F, t, T, f-;, F-;
- H, M, L
- zz, zt, zb

# 移动

- Ctrl-D, Ctrl-U
- Ctrl-F, Ctrl-B

# 移动

- `gg, G, ggV, ggD, ggyG`
- `gd` #goto declaration
- `gf` #goto file `Ctrl-^`
- `gi` #go to last edited location
- `gv` #reselect last visual selection
- `'` #jump back to last edited line

# 移动

- Ctrl-] #go to tag match
- Ctrl-O #go back jump history
- Ctrl-I #go forward jump history

# 移动

- Ctrl-W, Ctrl-W-H/K/J/L
- Ctrl-WR #swap window
- Ctrl-W= #resize
- Ctrl-WS #split window
- Ctrl-WV #vsplit window

# 编辑

- dw, dd, D
- yw, yy, p, P
- r, R
- o, O, i, I, a, A, s, S, c, C

# 编辑

- u, Ctrl-R
- J, :%j
- =
- >>, <<, :retab
- gu, gU, ~

# 编辑

- v,V,V-:!sort #选择
- Ctrl-V # 列操作
- Ctrl-V-I-Esc-Esc

# 编辑

- Ctrl-N, Ctrl-P #自动补全
- Ctrl-E, Ctrl-Y #复制前后一行的字符
- Ctrl-X Ctrl-K #字典补全
- Ctrl-X Ctrl-L #整行补全

# 搜索

- `/, ?, #, *`
- `n, N`
- `\c` 不区分大小写
- `\C` 强制区分大小写
- `\<, \>` 匹配单词边界

# 文件内容搜索

- `:grep string -r directory`  
`:grep DealHelper -r template/deal/`
- `:cw, cn, cp`

# 替换

- `:%s/old/new/g` 全局替换
- `:'<,>s/old/new/g` 区块替换
- `:g/^$/d` delete blank lines
- `:g/^/m0` reverse whole file
- `:'<,>g/^$/d`

# 文本对象

- dit, dat, di', di'', di(, dip, di{
- yit, yat, yi'....
- vit, vat, vi'

# 折疊

- zo, zc, zO, zC, zR, zM

:h folding

# 标签页

- `tabe filename`
- `gt, gT`
- `set showtabline=2`
- `nmap <C-N> :tabnext<CR>`  
`nmap <C-P> :tabprevious<CR>`

`:h tabpage.txt`

# 缓冲区

- :ls, b l ...9, bn, bp
- :cd, pwd

:h buffer-hidden

# 寄存器

- reg
- “lp
- “%p

# 位置标记

- $m\{a-zA-Z\}$
- 'a, 'g
- marks

:h mark-motions

# Ctags

- `$ ctags -R`
- `$ ptags -R`
- `Ctrl-]`, `Ctrl-O`, `Ctrl-I`

# 杂项

- `:set dictionary=file` 设置字典文件
- `K`
- `:sh` 启动一个子shell `Ctrl-D`返回
- `!:执行shell命令` `!:php -l %` `!:git diff %`

# 杂项

- `set paste` 粘帖代码
- `set nopaste` 停止粘帖代码
- `nohl`

# 配置

- 全局配置
  - `/etc/vimrc`
  - `/usr/share/vim/`
- 用户配置
  - `~/.vimrc`
  - `~/.gvimrc`
  - `~/.vim/`

# 用戶配置

- `.vimrc` - user settings
- `.vim/`
  - `after/` - loaded at the very end
  - `autoload/` - automatically loaded scripts
  - `colors/` - custom color schemes
  - `doc/` - plugin documentation
  - `ftdetect/` - filetype detection scripts
  - `ftplugin/` - filetype plugins
  - `indent/` - indent scripts
  - `plugin/` - plugins
  - `syntax/` - syntax scripts

# 常用设置

- `set nocp`
- `set ru`
- `set hls`
- `set is`
- `set number`
- `set wildmenu`
- `set enc=utf-8`

# 常用设置

- `set tabstop=4`
- `set shiftwidth=4`
- `set softtabstop=4`
- `set expandtab`
- `set autoindent`
- `syntax on`

# 键盘映射

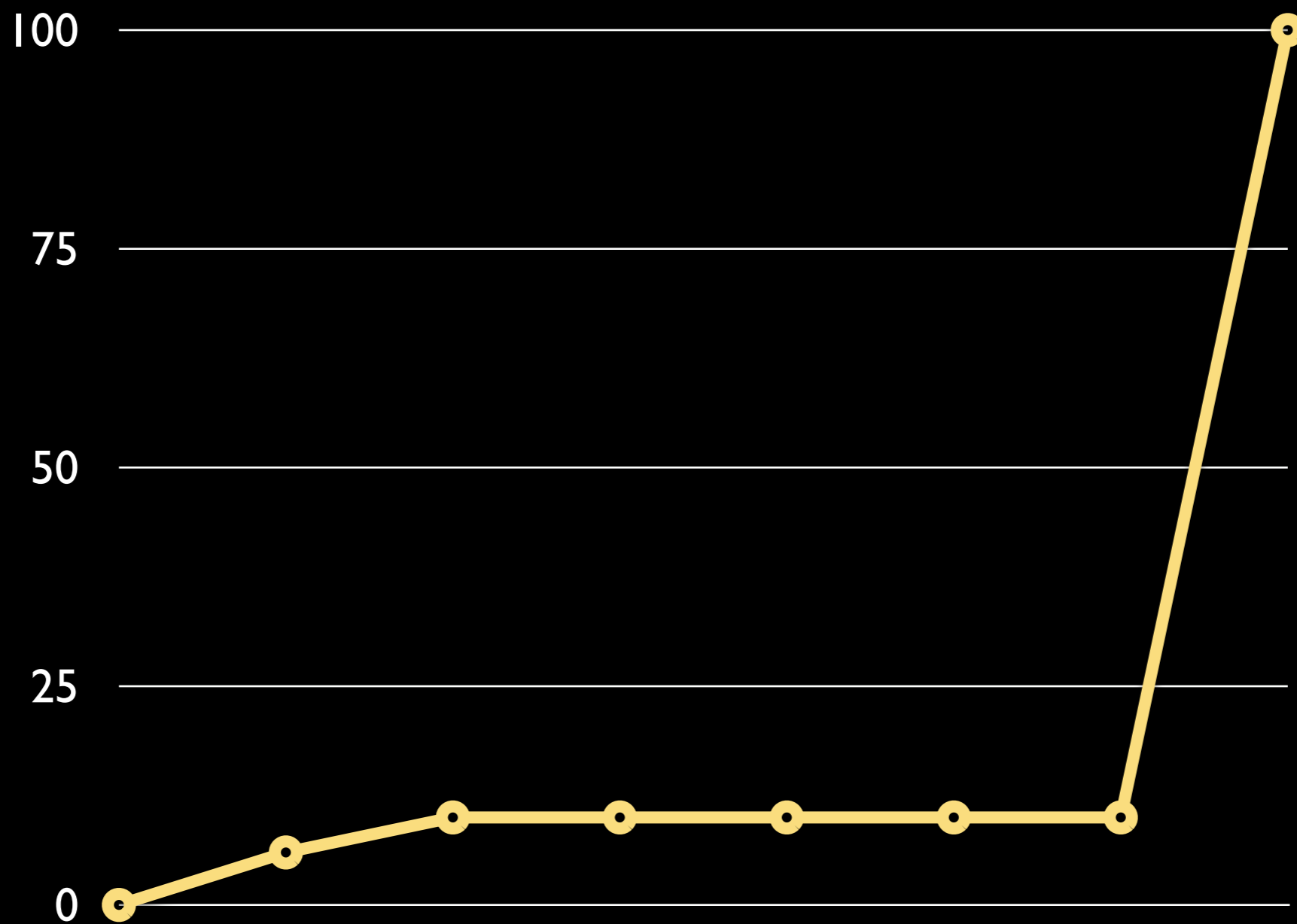
- map <F12> :set number!<CR>
- nmap <C-N> :tabnext<CR>
- nmap <C-P> :tabprevious<CR>

# 自动命令

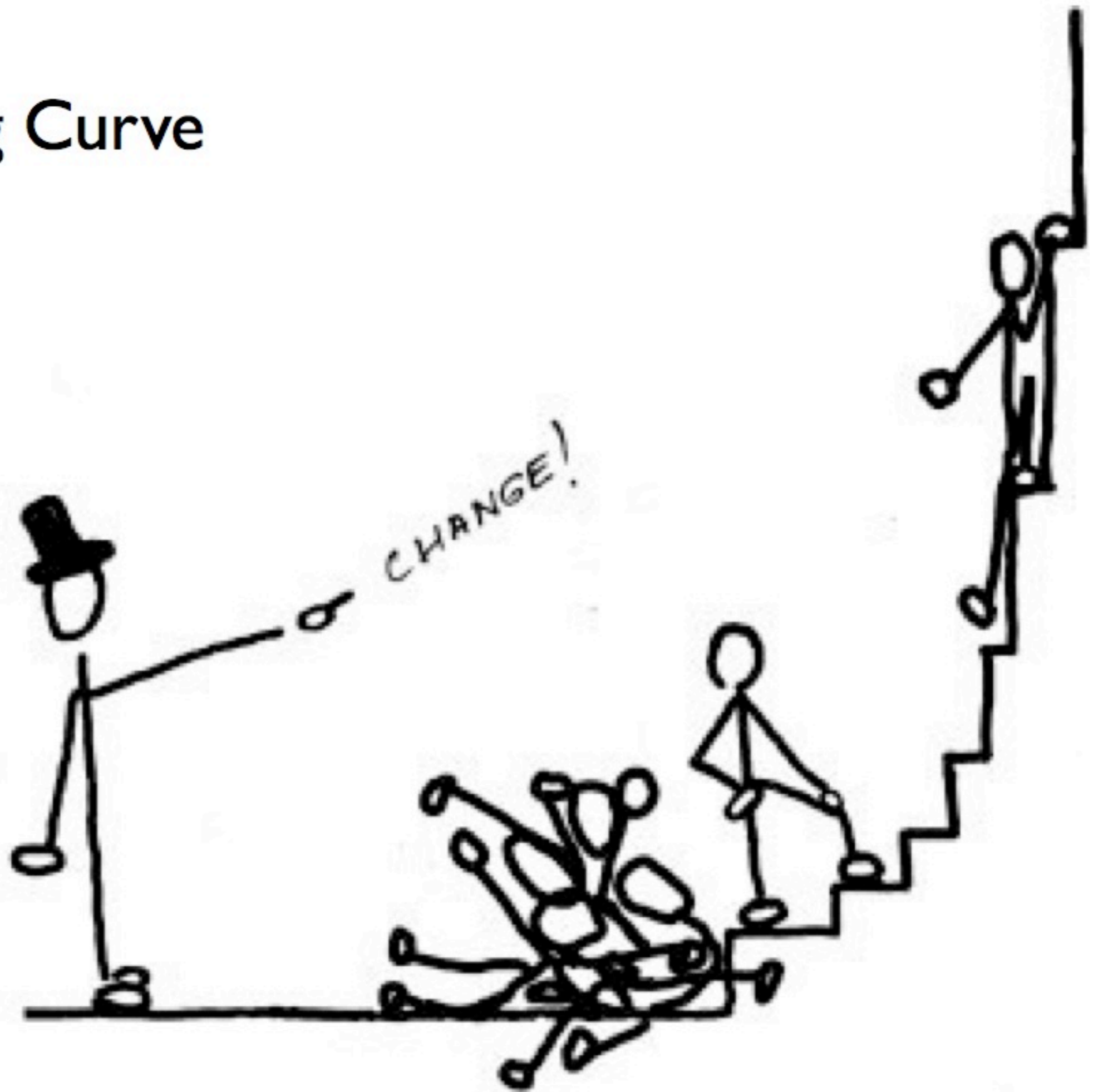
- `autocmd FileType javascript set makeprg=jsl\ -nologo\ -nofilelisting\ -nosummary\ -nocontext\ -process\ %`
- `autocmd FileType javascript set errorformat=%f (%l):\ %m`
- `autocmd FileType javascript inoremap <silent> <F9> <C-O>:make<CR>`
- `autocmd FileType javascript map <silent> <F9> :make<CR>`

# 学习的方法

# 陡峭的学习曲线



# Learning Curve







找一个好老师





精读手册





实践出真知

# 从哪里开始

```
$ vimtutor
```

# 个人哲学

# 爱Terminal 不爱GUI

- 结合screen和shell，超越IDE的愉悦体验

# 爱Terminal 不爱GUI

- 结合screen和shell，超越IDE的愉悦体验
- 在本地和服务器之间快速自由切换

# 爱默认配置 不爱花哨

- 尽量保持一致，避免坠入配置的地狱

# 爱默认配置 不爱花哨

- 尽量保持一致，避免坠入配置的地狱
- 减少插件依赖，不断尝试内部操作组合

# 爱VIM 也爱EMACS

- 骑墙派

# 爱VIM 也爱EMACS

- 骑墙派
- vim and emacs are everywhere

# 爱VIM 也爱EMACS

- 骑墙派
- vim and emacs are everywhere
- vim:less, slashdot, gmail, firefox, chrome, eclipse, visual studio, etc...

# 爱VIM 也爱EMACS

- 骑墙派
- vim and emacs are everywhere
- vim:less, slashdot, gmail, firefox, chrome, eclipse, visual studio, etc...
- emacs:bash, mac, etc...

# 心得

# 心得

- 攀爬学习曲线的一次性付出，得到的是更有效编写程序的能力

# 心得

- 攀爬学习曲线的一次性付出，得到的是更有效编写程序的能力
- 精力也可以更多地放在设计层面而不是低层次的细节操作

# 心得

- 攀爬学习曲线的一次性付出，得到的是更有效编写程序的能力
- 精力也可以更多地放在设计层面而不是低层次的细节操作
- 能力可能会暂时倒退，但蹲下来是为了跳得更高

VIM不能保证你成为优  
秀的程序员

# VIM不能保证你成为优秀的程序员

- 用刘国梁的球拍也不能保证成为冠军

# VIM不能保证你成为优秀的程序员

- 用刘国梁的球拍也不能保证成为冠军
- 没人会在乎贝多芬用什么钢琴，达芬奇用什么画笔

# VIM不能保证你成为优秀的程序员

- 用刘国梁的球拍也不能保证成为冠军
- 没人会在乎贝多芬用什么钢琴，达芬奇用什么画笔
- vim拥有非常务实的工具精神，它本身只是一个工具

# VIM不能保证你成为优秀的程序员

- 用刘国梁的球拍也不能保证成为冠军
- 没人会在乎贝多芬用什么钢琴，达芬奇用什么画笔
- vim拥有非常务实的工具精神，它本身只是一个工具
- 优秀的程序员是在不断思考和解决问题中成长起来的

# THANK YOU

- Q&A
- <https://github.com/panweizeng/env>
- [http://en.wikipedia.org/wiki/Lifted\\_\(2006\\_film\)](http://en.wikipedia.org/wiki/Lifted_(2006_film))