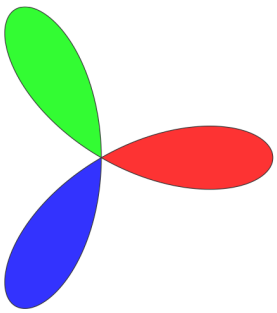
tmux 使用笔记

Unix/Linux 终端复用器

作者：凡云

时间：March 31, 2023

版本：第一版



工欲善其事必先利其器

目录

前言

tmux(terminal multiplexer) 是一个让人相见恨晚的终端复用器，虽然不直接参与 coding, 但却可以大大提高工作效率。如果你的工作主要是在远程服务器上进行, 那么它将是你的福音.

tmux 允许你在一个窗口中运行多个终端，而不必为每个终端都新开一个窗口. tmux 最强大的地方在于我们可以随时离开和加载当前会话. 每个会话都是持久性的, 即使存在意外让终端断开 (比如网络掉线等).

先安利一下为什么要用 tmux?

- 会话是持久性的, 可以随时退出和加载
- 支持打开多个窗口, 一个窗口可以再细分为子窗口
- 快捷键丰富, 远离鼠标, 操作效率高

同类工具还有 GNU Screen 和 Byobu.

第 1 章 tmux

1.1 安装

通常系统上默认是没有 tmux 工具的，需要手动安装。

- debian 系: sudo apt-get install tmux
- redhat 系: sudo yum install tmux
- Mac OS: brew install tmux

1.2 基本架构

tmux 采用了 CS (Client-Server) 架构; 但 tmux server 默认是没有启动的, 当第一个 client 启动时, 如果发现 Server 没有启动, 默认行为是启动一个 Server 进程后台执行; 当所有会话退出时, Server 进程退出. Client 和 Server 通过 Socket 通信. 实际测试, tmux v3.0 和 tmux v3.1 符合上述行为. 参考图 1.1

基本概念:

★ *会话 (Session)*, 一个大的伪终端集合, 用来管理窗口; 可以有一个或多个窗口

★ *窗口 (Window)*, 会话的基本元素, 仅有一个窗口展现给用户; 窗口可以再次切分为子窗口

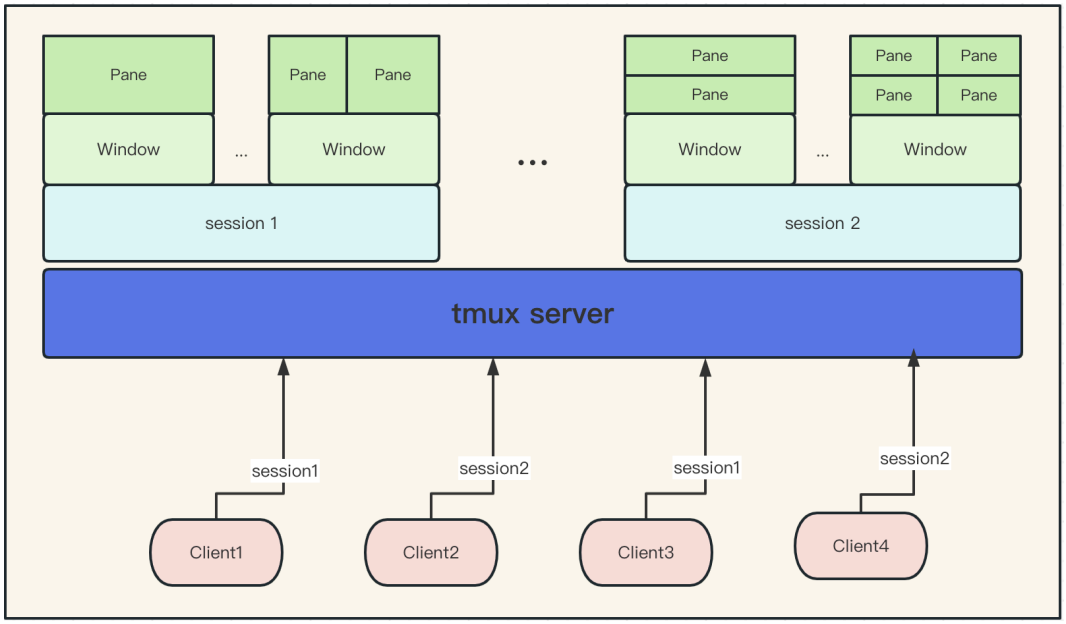
★ *子窗口 (Pane)*, 窗口的基本元素, 一个窗口可以有一个或多个子窗口; 每个子窗口运行一个伪终端

★ *client*, tmux 前端进程, 和一个会话关联 (attach) 在一起; 创建会话或者在后续通过 attach-session 命令加载会话时, 会启动 tmux client. 用户所看到的 Window 和 Pane, 都是 tmux client 展现出来的;

★ *server*, tmux 后台进程, 用来管理 client, Session, Window 和 Pane

★ *command*, tmux 支持的二级命令, 用来实现各种交互

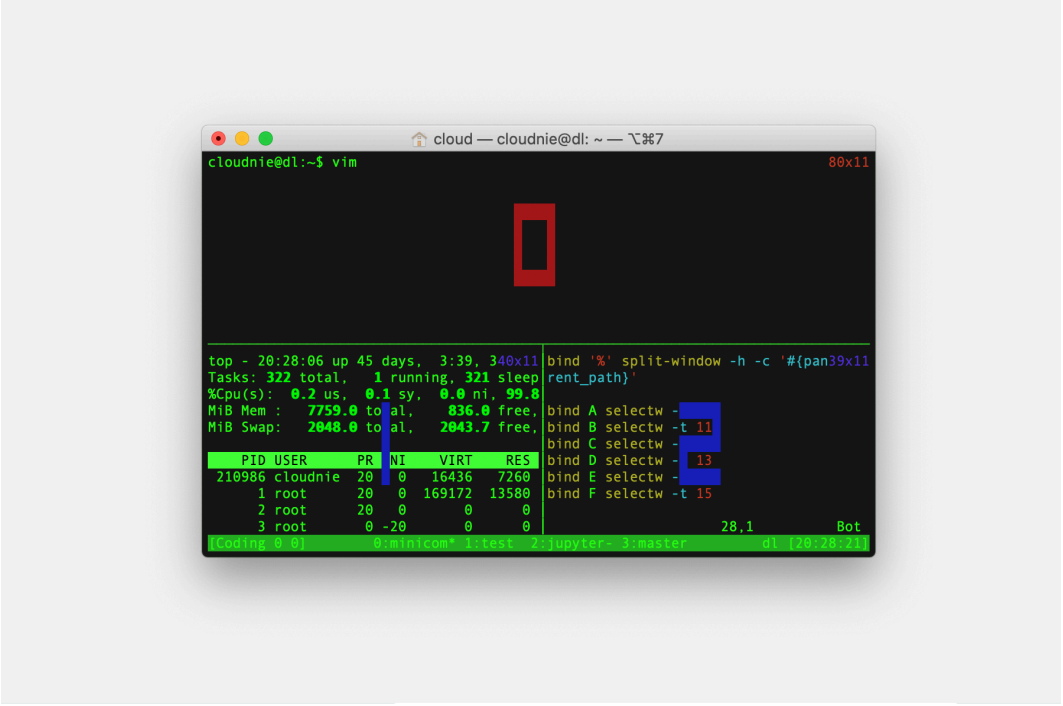
图 1.1: tmux 基本概念



基本用法: tmux [-2CluvV] [-c shell-command] [-f file] [-L socket-name] [-S socket-path] [command [flags]]

常用命令行选项介绍:

图 1.2: tmux 示例



★ *-f*, 指定备选配置文件, 替换系统默认配置文件
★ *-L*, 设定 tmux server 的 socket 名字, 使得同一个主机可以运行多个 tmux server
★ *-V*, 显示 tmux 版本号
★ *-C*, 运行在控制模式
★ *-S*, 设定 socket 路径, *-L* 参数会被忽略
★ *command*, 指定 tmux 要执行的命令; 如果没有指定 command, tmux 默认创建新的会话 (Session).

第 2 章 快捷键

在 tmux client 中, tmux 的命令可以通过快捷键来触发; 快捷键由前缀和命令键组成; 前缀通常为一对复合键, 比如 Ctrl + b (默认值), 或者 Ctrl + a, 可以自由设定; 命令键通常为一个单键, 当然也可以是复合键.

在 tmux 的命令或者配置文件中, 会对一些特殊键做使用约定, 如下:

- ★ Ctrl　　*约定写为 C- 或者 ^*
- ★ Alt　　*约定写为 M-*
- ★ Delete　　*约定写为 DC*
- ★ BackSpace　　*约定写为 BSpace*
- ★ Insert　　*约定写为 IC*

2.1 前缀键设定

tmux 的快捷键前缀默认是 Ctrl-b, 可以通过命令 set 来修改. 下面这段代码将前缀由默认值 Ctrl-b 修改为 Ctrl-a.

```
# tmux set -g prefix C-a
# tmux unbind C-b
# tmux bind C-a send-prefix
```

不过上述修改仅对当前 tmux server 有效; 在新的 tmux server 中是无效的. 如果期望永久生效, 可以将上述命令写入配置文件. 参考 Referencestmuxconfig

2.2 默认快捷键

tmux 默认将一系列快捷键和命令绑定在一起, 当个绑定关系可以通过 bind-key 和 unbind-key 命令修改; 如果需要永久性修改, 可以在配置文件中加入绑定命令.

常用默认快捷键如下 (假设前缀键为 Ctrl-b, 简写为 C-b):

- C-b ?　　*列出所有快捷键和命令的绑定关系*
- C-b $　　*重命名当前会话*
- C-b !　　*将当前子窗口 (Pane) 移动到一个新窗口 (Window)*
- C-b ”　　*当前子窗口 (Pane) 分为上下两个子窗口 (Pane)*
- C-b %　　*将当前子窗口 (Pane) 分为左右两个子窗口 (Pane)*
- C-b ’　　*根据窗口 ID 选择窗口 (Window)*
- C-b (　　*将当前 clinet 关联到上一个会话 (sessions)*
- C-b)　　*将当前 clinet 关联到下一个会话 (sessions)*
- C-b ,　　*重命名当前窗口 (Window)*
- C-b &　　*销毁当前窗口 (Window)*
- C-b 0-9　　*根据 ID 选择窗口 (Window)*
- C-b ;　　*移动到上一个活动子窗口 (Pane)*
- C-b D　　*从会话的关联 client 列表中选择一个踢退*
- C-b [　　*进入 copy 模式*
- C-b c　　*创建一个新窗口*
- C-b d　　*踢退当前 client*
- C-b i　　*显示当前窗口 (Window) 信息*

- C-b l　　跳回到上一次选择的窗口
- C-b n　　跳回到下一个窗口
- C-b o　　选择下一个子窗口
- C-b p　　跳回到上一个窗口
- C-b s　　为当前 *client* 交互式选择新的会话 *(Session)*
- C-b t　　显示当前系统时间
- C-b w　　交互式选择窗口
- C-b x　　销毁当前子窗口 *(Pane)*
- C-b z　　打开或关闭当前子窗口的全屏模式
- C-b {　　交换当前子窗口和上一个子窗口的位置
- C-b }　　交换当前子窗口和下一个子窗口的位置
- C-b Up　　选择上面子窗口 *(Pane)*
- C-b Down　　选择下面子窗口 *(Pane)*
- C-b Left　　选择左边子窗口 *(Pane)*
- C-b Right　　选择右边子窗口 *(Pane)*
- C-b Space　　重新排列当前窗口 *(Window)* 中的子窗口 *(Pane)*
- C-b C-Up　　向上调整子窗口
- C-b C-Down　　向下调整子窗口
- C-b C-Left　　向左调整子窗口
- C-b C-Right　　向右调整子窗口

2.3 修改快捷键

如果不习惯 tmux 默认的快捷键绑定关系, 通过 bind-key 和 unbind-key 命令可以自由修改快捷键和命令的绑定关系. 下面这段示例修改选择子窗口相关命令对应的快捷键.

```
# tmux bind-key 'u' select-pane -U
# tmux bind-key 'j' select-pane -D
# tmux bind-key 'k' select-pane -R
# tmux bind-key 'h' select-pane -L
```

参考: 6, 可以将这些命令写入配置文件, 达到持久性生效的目的.

第 3 章 常用命令

执行 tmux 命令有三种方式:

- 通过 shell 执行, 比如 *tmux selectw -t 9*, 选择窗口 9
- 通过快捷键执行, 比如 *C-b 9*, 选择窗口 9
- 命令提示符窗. 首先利用快捷键 *C-b :* 打开命令提示符, 然后输入待执行命令, 比如 *selectw -t 9*

本章节提到的 tmux 命令时, 上述三种方式都会时遇到. 以冒号 (:) 开始的命令代表在提示符窗口中执行的.

3.1 会话和客户端常用命令

1. **创建会话** *new-session, alias:new*

   创建一个名字为 *coding* 的会话并加载到当前终端:

```
# tmux new-session -s coding
```

   创建一个名字为 *sdk* 的会话但是不加载:

```
# tmux new-session -d -s sdk
```

   如果已经处于 tmux client 中, 则不能再次创建新的会话; 否则会得到以下警告.

```
# tmux
sessions should be nested with care, unset \$TMUX to force
```

   在 shell 中运行 tmux 并且不给任何参数, tmux 默认会创建一个会话并加载.

2. **加载会话** *attach-session, alias:attach*

```
# tmux attach -t coding
```

   上述命令加载会话 *coding*; 如果已经处于 tmux client 中, 则会报以下 warning.

```
# tmux attach -t test
sessions should be nested with care, unset \$TMUX to force
```

3. **查看当前主机已创建的会话** *list-sessions, alias:ls*

```
# tmux list-sessions
coding: 1 windows (created Sat Apr 1 16:16:11 2023)
system: 3 windows (created Sat Apr 1 16:17:25 2023)
```

4. **查看 client 列表** *list-clients, alias:lsc*

```
# tmux list-clients
/dev/pts/0: coding [204x61 xterm-256color] (utf8)
/dev/pts/10: system [204x61 xterm-256color] (utf8)
/dev/pts/11: coding [204x61 xterm] (utf8)
```

   通过上述命令的结果, 我们可以看到, 会话 *coding* 被加载 (attach) 了两次, 分别对应两个 client.

5. **退出 client** *detach-client, alias:detach*

   例: 退出当前 client *:detach-client* 快捷键: *C-b d*

   例: 退出 client /dev/pts/21 *:detach-client -t /dev/pts/21* 快捷键: *C-b D*

   例: 退出会话 sdk 的所有 client *:detach-client -s sdk*

6. **销毁 tmux server** *kill-server*
   kill-server 除了销毁 tmux server, 还会销毁所有关联 client 和会话.
7. **销毁 tmux 会话 (Session)** *kill-session*
   例: 销毁当前会话 *:kill-session*
   例: 销毁会话 coding *:kill-session -t coding*
   kill-session 执行后, 和会话关联的 client 会被强制踢退, 关联的窗口, 子窗口也会被销毁.
8. **重命名当前会话 (Session)** *rename-session, alias:rename*
   例: 将当前会话重命名为 coding *:rename coding*
   例: 将会话 test2 重命名为 test3 *:rename -t test2 test3*
9. **重新选择 client 关联的会话** *switch-client, alias:switchc*
   下述命令的作用是将 client /dev/pts/21 的会话由 *system* 切换为 *test3*:

```
# tmux list-sessions
system: 4 windows (created Wed Apr 5 21:30:57 2023) (attached)
test3: 1 windows (created Wed Apr 5 21:16:29 2023)
# tmux list-clients
/dev/pts/21: system [204x61 xterm-256color] (utf8)
/dev/pts/0: system [204x61 xterm-256color] (utf8)
# tmux switch-client -c /dev/pts/21 -t test3
```

10. **列出 tmux 所有命令** *list-commands*
11. **列出 tmux 所有绑定的快捷键** *list-keys*

3.2 窗口和子窗口常用命令

如果已经处于 tmux client 上下文, 那么操作窗口和子窗口将是常见的任务. tmux 定义了一系列命令来操作窗口和子窗口, 参考如下 (通常有快捷键对应的属于高频命令):

1. **创建窗口 (Window)** *new-window, alias:neww*
   例: 创建窗口 *:new-window* 快捷键: *C-b c*
   例: 创建名字为 test 的窗口 *:new-window -d -n test*
   例: 创建窗口, 并执行 top 命令 *:new-window -n master top*
   在指定会话中创建窗口, 下述命令在会话 *sdk* 中创建一个窗口命名为 *maintain*, 工作目录为 */media/data*:

```
# tmux neww -d -c /media/data -n maintain -t sdk
```

   -d 的作用是指示当前会话的活动窗口维持不变; 默认行为是切到新建窗口.
   -c 的作用是设定 shell 工作目录.
2. **重命名窗口** *rename-window alias:renamew*
   例: 将当前窗口重命名为 master *:renamew master* 快捷键: *C-b ,* 输入: *master*
   在 tmux client 上下文环境中, 推荐使用快捷键 *C-b ,* 重命名窗口.
   下述命令将会话 *sdk* 的窗口 0 重命名为 *minicom*:

```
# tmux rename-w -t sdk:0 minicom
```

   下述命令将会话 *sdk* 的窗口 minicom 重命名为 *console*:

```
# tmux rename-w -t sdk:minicom console
```

3. **选择窗口 (Window)** *select-window alias: selectw*

   例: 选择窗口 0 *:selectw -t 0* 快捷键: *C-b 0*

   例: 选择窗口 9 *:selectw -t 9* 快捷键: *C-b 9*

   例: 选择窗口 10 *:selectw -t 10* 快捷键: *C-b '* 输入: *10*

   例: 选择前面一个窗口 *:selectw -p* 快捷键: *C-b p*

   例: 选择后面一个窗口 *:selectw -n* 快捷键: *C-b n*

   例: 选择上次使用的窗口 *:selectw -l* 快捷键: *C-b l*

   注意, 命令 previous-window, next-window, last-window 分别等价于 selectw -p, selectw -n, selectw -l.

4. **移动窗口** *move-window, alias:movew*

   下述命令将 ID 为 12 的窗口移动到 ID 1, 前提是 ID 1 是空置状态:

```
# tmux movew -s 12 -t 1
```

5. **显示窗口 (Window) 信息** *list-windows alias:lsw*

```
# tmux lsw -a
coding:0: minicom (1 panes) [204x60]
coding:1: master (1 panes) [204x60]
coding:2: maintain (1 panes) [204x60]
sdk:0: console- (1 panes) [80x24]
sdk:1: master* (1 panes) [80x24]
```

6. **销毁窗口 (Window)** *kill-window*

   下述命令将销毁当前会话的命名为 *customer* 的窗口:

```
# tmux kill-window -t customer
```

   下述命令将销毁会话 *sdk* 的窗口 *console*

```
# tmux kill-window -t sdk:console
```

7. **分隔子窗口** *split-window, alias:splitw*

   例: 水平分隔当前子窗口 *:split-window -h* 快捷键: *C-b %*

   例: 垂直分隔当前子窗口 *:split-window -v* 快捷键: *C-b "*

   下述命令, 将水平分隔会话 *coding* 的窗口 0, pane 0:

```
# tmux split-window -h -t coding:0.0
```

   下述命令, 将垂直分隔会话 *coding* 的窗口 1, pane 2:

```
# tmux split-window -h -t coding:1.2
```

8. **选择子窗口 (Pane)** *select-pane, alias:selectp*

   例: 选择左侧子窗口 *:select-pane -L* 快捷键: *C-b Left*

   例: 选择右侧子窗口 *:select-pane -R* 快捷键: *C-b Right*

   例: 选择上方子窗口 *:select-pane -U* 快捷键: *C-b Up*

   例: 选择下方子窗口 *:select-pane -D* 快捷键: *C-b Down*

   例: 选择上一次使用的子窗口 *:select-pane -l* 快捷键: *C-b ;*

   例: 选择上一次使用的子窗口 *:last-pane -l* 快捷键: *C-b ;*

   下述命令, 将会话 *coding* 窗口 2 的活动子窗口设定为 3 号子窗口:

```
# tmux select-pane -t coding:2.3
```

9. **关闭或打开子窗口的输入** *select-pane, alias:selectp*

   下述命令, 将关闭子窗口 0 的输入, 该子窗口位于会话 *coding* 的窗口 2:

```
# tmux select-pane -d -t coding:2.0
```

   下述命令, 将打开子窗口 0 的输入, 该子窗口位于会话 *coding* 的窗口 2:

```
# tmux select-pane -e -t coding:2.0
```

10. **重命名子窗口 (Pane)** *select-pane*

    下述命令, 将子窗口 0 的 Title 设定为 *10.0.14.16*, 该子窗口位于会话 *coding* 的窗口 2:

```
# tmux select-pane -T 10.0.14.16 -t coding:2.0
```

11. **更改子窗口 (Pane) 的大小** *resize-pane*

例: 向左调整子窗口, step=1	*:resize-pane -L 1*	快捷键: *C-b C-Left*
例: 向左调整子窗口, step=5	*:resize-pane -L 5*	快捷键: *C-b M-Left*
例: 向右调整子窗口, step=1	*:resize-pane -R 1*	快捷键: *C-b C-Right*
例: 向右调整子窗口, step=5	*:resize-pane -R 5*	快捷键: *C-b M-Right*
例: 向上调整子窗口, step=1	*:resize-pane -U 1*	快捷键: *C-b C-Up*
例: 向上调整子窗口, step=5	*:resize-pane -U 5*	快捷键: *C-b M-Up*
例: 向下调整子窗口, step=1	*:resize-pane -D 1*	快捷键: *C-b C-Down*
例: 向下调整子窗口, step=5	*:resize-pane -D 5*	快捷键: *C-b M-Down*
例: 进入或退出 zoom 状态	*:resize-pane -Z*	快捷键: *C-b z*

12. **选择子窗口 (Pane) 布局** *select-layout, alias:selectl*

例: 选择水平布局	*:select-layout even-horizontal*	快捷键: *C-b M-1*
例: 选择垂直布局	*:select-layout even-vertical*	快捷键: *C-b M-2*
例: 选择 main-horizontal 布局	*:select-layout main-horizontal*	快捷键: *C-b M-3*
例: 选择 main-vertical 布局	*:select-layout main-vertical*	快捷键: *C-b M-4*
例: 选择 tiled 布局	*:select-layout tiled*	快捷键: *C-b M-5*

13. **选择上一个子窗口布局** *previous-layout, alias:prevl*
14. **选择下一个子窗口布局** *next-layout, alias:nextl* 快捷键: *C-b Space*
15. **选择上次使用的子窗口布局** *last-layout, alias:lastl*
16. **销毁子窗口** *kill-pane* 快捷键: *C-b x*
17. **显示子窗口 ID** *display-panes* 快捷键: *C-b q*
18. **移动子窗口** *move-pane, alias:movep*
19. **显示子窗口信息** *list-panes, alias:lst*

```
# tmux lsp
0: [204x23] [history 0/2000, 0 bytes] %5
1: [204x13] [history 65/2000, 25205 bytes] %13 (active)
2: [204x22] [history 59/2000, 26798 bytes] %16
```

3.3 选项 (Option) 设定命令

命令 set-option 可以用来设定 tmux server, 会话和窗口的属性选项, 别名: set. 命令 show-options 可以用来查看选项设定, 别名: show.

基本用法: set-option [-aFgoqsuw] [-t target-session | target-window] option value

参数介绍:

★ *-s* 设定 tmux server 属性选项
★ *-w* 设定窗口属性选项, 等同于 set-window-option
★ *-g* 设定会话和窗口的全局属性选项, s 对 erver 内的所有会话和窗口生效
★ *-u* 取消一个选项的设定
★ *-a* 给选项追加内容, 而不是覆盖
★ *-F* 待设定值使用特殊变量
★ *-t* 指定目标会话或窗口

一些常用选项:

1. **command-alias** *例: 设定 list-session 的别名为 lss* *: set -s command-alias lss='list-sessions'*
2. **base-index** *例: 设定窗口 ID 从 1 开始* *: set -g base-index 1*
3. **prefix** *例: 设定前缀键为 C-a* *: set -g prefix C-a*
4. **prefix2** *例: 设定备选前缀键为 M-b* *: set -g prefix2 M-b*
5. **status** *例: 关闭当前会话的状态栏* *: set status off*
6. **status-interval** *例: 刷新状态栏的周期为 2 秒* *: set status-interval 2*
7. **status-left** *例: 状态栏左侧显示会话名字* *: set -g status-left '#S'*
8. **status-right** *例: 状态栏右侧显示 uptime 结果* *: set -g status-right '#(uptime)'*
9. **status-position** *例: 设置状态栏显示上方* *: set -g status-position top*
10. **mouse** *例: 关闭支持鼠标* *: set -g mouse off*

3.4 命令别名和简写

除了支持命令别名外, tmux 还支持命令简写, 可以只给出命令的前半部分, 比如 attach-session 的别名 attach, 可以简写为 a. 当 tmux 对输入的简写命令无法准确定为时, 会给出提示, 让我们进一步补全.

```
# tmux list
ambiguous command: list, could be: list-buffers, list-clients, list-commands, list-keys, list-panes,
    list-sessions, list-windows
# tmux list-c
ambiguous command: list-c, could be: list-clients, list-commands
# tmux list-w
0: bash* (1 panes) [204x60] [layout d888,204x60,0,0,29] @15 (active)
# tmux list-s
0: 7 windows (created Sun Apr 2 21:30:23 2023)
3: 1 windows (created Wed Apr 5 10:40:22 2023)
test: 3 windows (created Mon Apr 3 22:32:45 2023)
```

第 4 章 拷贝模式 (Copy Mode)

在介绍如何 copy 内容之前, 我们需要先了解 tmux 的 copy mode 和 buffer 机制.

4.1 Mode

tmux 的 window 有两种 mode, 分别是:

★ 默认 mode, 可以直接访问终端, 和 shell 交互; 可以看到 Window 最新的内容, 但无法看到历史内容.

★ copy mode, 允许窗口的当前部分部内容或历史内容被 copy 到 tmux buffer, 方便后续使用. 可以通过命令 *copy-mode* 或快捷键 *C-b [* 进入 copy mode.

在 copy mode 下, 操作文本 (选择/查找/跳转等) 有两种风格, 分别是 vi 风格和 emacs 风格. 默认是 emacs 风格, 可以通过窗口选项 mode-keys 来选择不同的风格.

设定为 VI 风格:

```
: set -g mode-keys vi
```

在不同的风格下, 操作文本的键值不一样, 比如在 vi 风格下, 通过字母键 *h,j, k, l* 移动光标; 但在 emacs 风格下, 分别是通过组合键 *C-b, C-f, C-n, C-p* 来移动光标.

参考 6, 可以将上述命令写入配置文件以实现持久性生效.

4.2 Buffer

tmux 内部维护了一个命名 buffer 列表, 并定义了一系列相关操作命令:

1. **从列表中选择 buffer** *choose-buffer* 快捷键: *C-b =*
2. **删除 buffer** *delete-buffer*
3. **列印 buffer** *list-buffers*
4. **从文件中加载 buffer** *load-buffer*
5. **保存 buffer 到文件** *delete-buffer*
6. **设定 buffer 内容** *set-buffer*
7. **显示 buffer 内容** *show-buffer*

4.3 如何拷贝和粘贴文本

在 tmux 环境中, 拷贝和粘贴文本需要分两个步骤, 首先进入 copy mode, 选择窗口内容并放到 buffer 列表; 然后退出 copy mode, 从 buffer 列表中选择指定的 buffer, 并执行粘贴动作.

在 vi 和 emacs 两种风格下, 选择窗口内容并放到 buffer 列表, 对应的快捷键不一样, 下面以 vi 风格为例介绍如何拷贝和粘贴窗口内容:

1. 通过快捷键 *C-b [* 或命令 *:copy-mode* 进入 copy Mode
2. 通过快捷键 *V* 开始选择内容, 通过 *h*, *j*, *k*, *l* 控制选择内容
3. 通过快捷键 *C-j* 或者 *Enter* 将选中内容放入 buffer 列表并退出 copy mode
4. 通过快捷键 *C-b]* 粘贴最近入队列的 buffer, 或者快捷键 *C-b =* 选择 buffer 并粘贴

第 5 章 状态栏

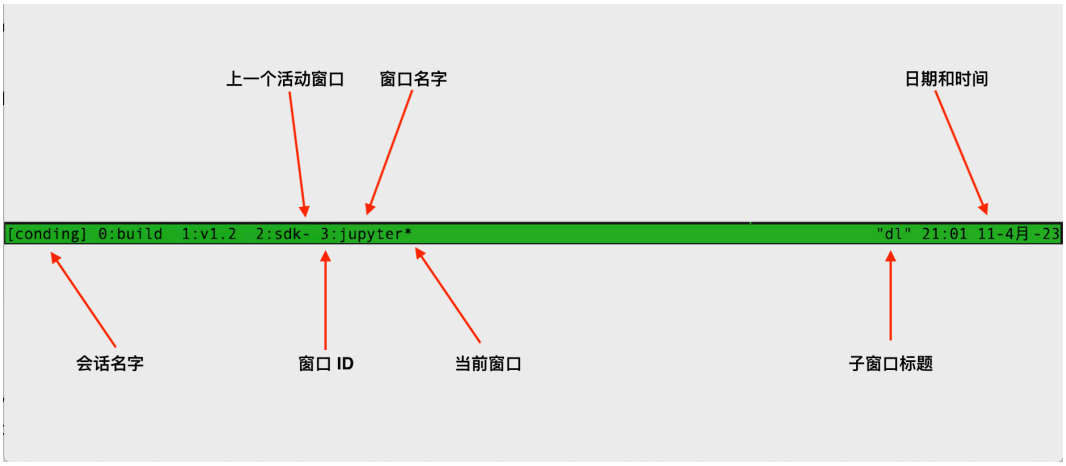
tmux 的状态栏或状态行默认位于终端界面的下方, 用来显示会话, 窗口和子窗口的信息, 通常包含三部分:

★ 左侧部分, 显示会话名字

★ 右侧部分, 显示子窗口标题, 日期和时间等.

★ 中间部分, 显示窗口信息, 包括窗口 ID 和名字. 名字后面会追加特殊符合用来标记窗口状态.

图 5.1: tmux 状态栏



有三个命令会直接影响状态栏的内容, 分别是 *command-prompt, confirm-before, display-message*.

5.1 定制状态栏

1. 设定状态栏背景色为红色 *:set -g status-style bg=red*
2. 窗口列表居中显示 *:set -g status-justify centre*
3. 每秒更新一次状态栏 *:set -g status-interval 1*
4. 左侧部分长度设定为 20 个字符 *:set -g status-left-length 20*
5. 左侧显示会话名字, 活动窗口 ID, 活动子窗口 ID *:set -g status-left '[#S #I #P]'*
6. 右侧显示活动子窗口标题和系统时间 *:set -g status-right '#T [%H:%M:%S]'*

如果期望上述修改持久生效, 需要将对应命令写入配置文件中. 参考: 6

第 6 章 配置文件

tmux 的一些默认设定不一定符合我们的习惯和期望, 可以根据需求来修改. 这些修改要放到 tmux 配置文件中.

配置文件的内容是一系列 tmux 自定义命令, 加载时会被依次执行. 这些命令可以参考: 3. tmux server 启动时会按照顺序从 */etc/tmux.conf* 和 *7.tmux.conf* 加载配置文件. 通常可以修改 *7.tmux.conf* 达到定制化的目的.

6.1 状态栏的修改

参考 5.1, 我们可以将相关命令写入配置文件 *7.tmux.conf*, 如下:

```
set -g status-left-length 20
set -g status-right '#T [\%H:\%M:\%S]'
set -gs escape-time 1
set -s status-style 'fg=red, bg=black'
set -g status-left '[#S #I #P]'
```

6.2 快捷键修改

除了新建和加载会话, 我们应该尽量避免使用 shell 或命令提示符的方式执行 tmux 命令, 尽可能通过快捷键来实现. 当默认快捷键不能满足我们的需求时, 可以新增或者重新映射快捷键. 原则是高频命令优先.

笔者根据自己的需求做了以下调整:

1. 重新映射选择子窗口的快捷键. tmux 默认的快捷键用到了 4 个方向键; 好处是容易记忆, 坏处是方向键通常位于键盘右侧, 势必会导致右手离开定位键 (H), 影响输入效率. 笔者使用了 *h, j, k, l* 来代替方向键.

   命令: 选择左侧子窗口 默认:*C-b Left* 新: *C-b h*
   命令: 选择上方子窗口 默认:*C-b Up* 新: *C-b j*
   命令: 选择下方子窗口 默认:*C-b Down* 新: *C-b k*
   命令: 选择右侧子窗口 默认:*C-b Right* 新: *C-b l*
   命令: 选择上一个窗口 默认:*C-b l* 新: *C-b C-l*

   原来 (C-b l) 映射到选择上一个使用窗口, 在这里我们选择一个代替: *C-b C-l*.

2. 子窗口大小的调整, tmux 默认选择了前缀键加上 Alt + 方向键, 同样影响输入效率; 笔者使用了 *H, J, K, L* 来代替方向键.

   命令: 向左侧调整子窗口大小 默认:*C-b M-Left* 新: *C-b H*
   命令: 向左侧调整子窗口大小 默认:*C-b M-Up* 新: *C-b J*
   命令: 向左侧调整子窗口大小 默认:*C-b M-Down* 新: *C-b K*
   命令: 向左侧调整子窗口大小 默认:*C-b M-Right* 新: *C-b L*

3. 窗口 ID 超过 10 后, 默认没有直接快捷键, 需要通过 *C-b '* 来快速输入 ID. 因为笔者在开发过程中, 打开的窗口比较多, 没有直接映射会影响效率, 因此笔者将 *C-b A-F* 映射到了选择窗口 10-15.

   命令: 选择窗口 10 默认:*C-b '* 新: *C-b A*
   命令: 选择窗口 11 默认:*C-b '* 新: *C-b B*
   命令: 选择窗口 12 默认:*C-b '* 新: *C-b C*
   命令: 选择窗口 13 默认:*C-b '* 新: *C-b D*
   命令: 选择窗口 14 默认:*C-b '* 新: *C-b E*
   命令: 选择窗口 15 默认:*C-b '* 新: *C-b F*

命令: 退出其他 client　　默认:*C-b D*　　新: *C-b X*

原来 *C-b D* 映射的命令是退出其他 client, 改为 *C-b X* 代替.

4. 子窗口水平和垂直分隔, tmux 默认使用的是 *C-b %* 和 *C-b "*, 但默认工作目录会切回 HOME 目录, 不能维持原来的工作目录. 笔者做了一个调整, 使用 *-c* 参数和特殊变量 *pane_current_path* 继承了待分隔子窗口的工作目录.

```
: bind '"' split-window -c '#{pane_current_path}'
: bind '\%' split-window -h -c '#{pane_current_path}'
```

6.3 其他

1. mode-keys 由 emac 调整为 vi

```
: set -gw mode-keys vi
```

2. 设定子窗口边框的背景色为黄色, 前景为红色

```
: set -g pane-active-border-style 'fg=red, bg=yellow'
```

第 7 章 其他

7.1 FORMATS

很多 tmux 命令可以接受一个 –F 的参数, 使得 tmux 可以扩展一些新的功能. 包括引用特殊变量, 更改输出内容, 变量比较, 变量替换, 数学计算, 匹配和搜索以及执行 shell 命令等.

请参考: Formats

7.2 样式 (SYTLE)

tmux 提供了 sytle 属性, 用来设置背景颜色, 前景颜色, 粗体, 斜体等.

常用属性:

1. bg 背景颜色
2. fg 前景颜色
3. bold 粗体
4. dim 变淡
5. underscore 增加底线
6. blink 闪烁
7. hidden 隐藏
8. italics 斜体
9. strikethrough 删除线

下面的命令, 修改状态栏右侧的 style, 前景颜色为白色, 粗体且闪烁:

```
: set status-right-style 'fg=white bold  blink'
```

详细用法参考 man

7.3 钩子 (hooks)

tmux 允许一个命令执行完成之后, 触发另一个命令的执行, 称之为钩子 (hooks). 可以通过 set-hooks 和 show-hooks 来设定和查看 hooks. 详细用法参考 man