Документация по встрече по консольным утилитам часть 2

Функции fishshell которые затронули

Функции на исходный текст которых можно посмотреть. Если понравилось нужно скопировать fish файл в ~/.config/fish/functions. Либо funced -s <ваше имя для функции без .fish> и вставить текст в появившийся редактор. Тогда функция будет с нужным вам именем.
После этого можно будет редактировать через funced -s <имя функции>
current_activity - показывает верхнюю активити текущего подключеного телефона или эмулятора
list-packages - список пакетов установленных приложений (просто распечатка)
fzf-list-packages - (использует list-packages) выбор одного пакета через fzf. Примеры использования:
- `adb uninstall (fzf-list-packages)` - удалить приложение выбранное в списке
- `adb shell pm clear (fzf-list-packages)` - очистка данных выбранного приложения
- fzf-list-packages | pbcopy - скопировать имя пакета в буфер обмена mac os x
Общий принцип. Все что в скобках это команда. Результат выполнения этой команды будет подставлен вместо скобок и результат выполнен. Аналогично в

bash $(command).
Для написания fishshell функций можно использовать https://openai-proxy.tcsbank.ru/ui/. В fishshell лучше всего умеет ChatGPT 4.

Плагин franciscolourenco/done

Показывает Mac уведомление если выполнение консольной команды заняло дольше чем 5 (по умолчанию, настраивается) секунд.
Установка `fisher install franciscolourenco/done` показывает стандартное уведомление если выполнение команды заняло более 5 секунд и в момент ее завершения терминал не был активным окном. Чтобы работало нужно разрешить уведомления от терминала в настройках мака.

Интеграция fishshell + Kitty

Позволяет просматривать вывод консольных команд через less (искать и листать вверх вниз).
Работает только с Kitty. (Для других терминалов ищите shell integration).
Нажимаем ⌘+, (стандатное сочетание для входа в настройки) или открываем в любимом редакторе файл ~/.config/kitty/kitty.conf
Читаем содержимое и правим по вкусу. Минимум рекомендую увеличить `scrollback_lines 200000` это количество строк которое терминал кэширует в памяти

для скрола. Чем больше тем больше терминал начинает кушать памяти после разрастания истории во вкладке. 1Gb+ с кучей старых вкладок - легко.
Далее прописываем внизу:

```
shell_integration enabled

scrollback_pager less -R

map ctrl+shift+z scroll_to_prompt -1

map ctrl+shift+x scroll_to_prompt 1

map ctrl+shift+h show_scrollback

map ctrl+shift+g
show_last_visited_command_output
```

Перезапускаем Kitty.
Объяснения.
shell_integration - включает интеграцию с fishshell. Теперь Kitty знает в какой момент началась и закончилась какая команда.
scrollback_pager less -R - команда используемая для пейджинга (-R позволяет сохранить подсветку синтаксиса)
Все команды которые можно забиндить на клавиши описаны в документации.
Конкретно этот вариант мапит ^ + ⇧ + z / x для

перехода к началу вывода любой из прошлых команд (z вверх и когда ушли вверх чтобы вернутся ниже x).
^ + ⇧ + g - отправляет вывод команды в less -R что позволяет его листать и искать в нем. Подробности аналогично любой другой команде `man less`.
^ + ⇧ + h - поиск по общей истории в буфере без привязки к конкретной команде.

Краткая инструкция по использованию `less`

Нажатие на `/` включает подсказку для поиска вниз от текущей точки. `?` - вверх. Можно писать любую регулярку (POSIX). После нажатия Enter включает подсветку найденного и дальше навигация N/P (next / previous).
Листаем традиционно: стрелки, PgUp/PgDn. Где-то в инструкции есть другие сочетания для PgUp/Dn если у вас их нет на клавиатуре (у меня есть, я их не знаю)).
Выход `q` 😂
Рекомендую поставить максимально доступную версию less через `brew install less`. Та что в маке может не содержать последних фиксов.

Другие команды

bat - подсветка синтаксиса для любого файла

`brew install bat` - подсветка синтаксиса для любого файла. Смотрит по расширению файла. Умеет в кучу

языков. Можно добавлять свои (Java, Kotlin, Python etc из коробки). После установки можно прописать `alias -s cat=bat`. Тогда вызов `cat` будет автоматом подсвечивать синтаксис для всех файлов. Конкретно у меня используется с ключами `bat --paging=never --style=snip`. Умеет в темы. Подробности в доках.

tldr - саммари по использованию команды (too long don't read)

`brew install tldr` - кратка выжимка по ключам команды. Антипод `man`. Например сравните:
`man ls` и `tldr ls`

lazygit - консольный гуй для гита

brew install lazygit - консольная тула по работе с git. Если запускать из папки репозитория показывает его содержимое. Запоминает репозитории с которыми работал. Позволяет переключать их при активном 1, 2 окне через ^ + R.
Доки. Тула быстрая и удобная (терпимо даже на репе банка, затыки не дольше 10 - 15 секунд, чаще всего с анимацией), не до конца френдли (маловато доков и они не всегда отражают текущий статус). Есть статья на Habr и видео на ютубе от автора на английском. Видео слегка устарело, скорее хвастовство чем туториал, но можно быстро увидеть что тула умеет.

Тем не менее это лучший консольный гуй для гита что мне попадался. Активно развивается.

delta - специализированный пейджер для патч файлов гита

`brew install delta` - специализированный пейджер для патч файлов гита. Раскрашивает дельты в файлах с учетом синтаксиса языка. Использует подсветку синтаксиса и темы от `bat`. Умеет работать с lazygit. Чтобы тема была общая с `bat` надо ее прописывать в переменную окружения - `BAT_THEME`, например, в файле `~/.config/fish/config.fish` так: `set -x BAT_THEME "Catppuccin Macchiato"`.
Конкретно тема капучино скачивается отдельно и устанавливается. Есть темы из коробки `bat --list-themes`. Можно просто искать `bat themes` на Github или в Google.
`delta` настраивается целиком через переменные окружения или параметры командной строки. Чтобы настраивать через конфиг надо передавать путь к конфигу в настройках.

Настройка для консольной команды `git` (diff, show, log, etc)

`git config --global --edit` открывается редактор с глобальным гит конфигом или открываем файл любимым редактором ~/.gitconfig. Добавляем:

```
[core]
        pager = delta --config
~/.config/delta/delta.config
... другие настройки в секции если есть
[interactive]
        diffFilter = delta --config
~/.config/delta/delta.config --color-only
```

Настройка для lazygit.

После входа в `lazygit` нажимаем `1` (переход в первое окно [1] Status). Нажимаем 'e' открывается редактор с конфигом или открываем его в любимом редакторе тут `~/Library/Application Support/lazygit/config.yml`. Добавляем туда с учетом синтаксиса yaml

```
git:
  paging:
    colorArg: always
    pager: delta --config
~/.config/delta/delta.config --dark --
paging=never
```

Перезапускаем lazygit.
Там же рядом лежит файл `state.yml` в котором лежат настройки UI. Из интересного туда можно добавить в `recentrepos` пути к репозиториям и переключать их потом через ^ + R без выхода из программы.

Пример моего конфига для дельты тоже есть в архиве (положить в `~/.config/delta/delta.config` если использовать как описано выше). Подробности по ключам в документации