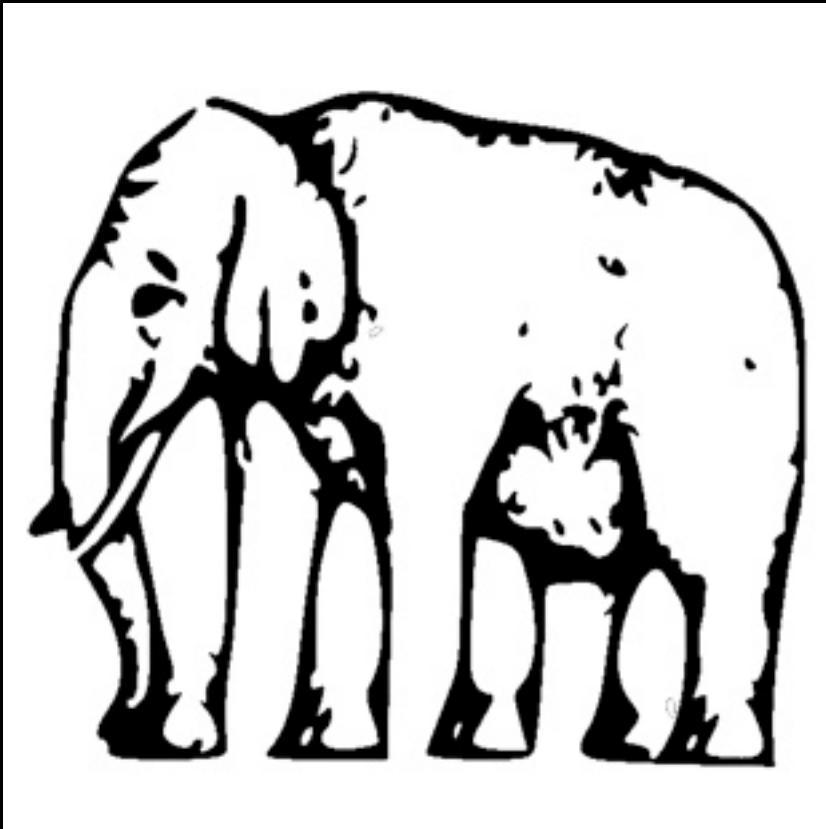


# I've been looking for freedom 🎶

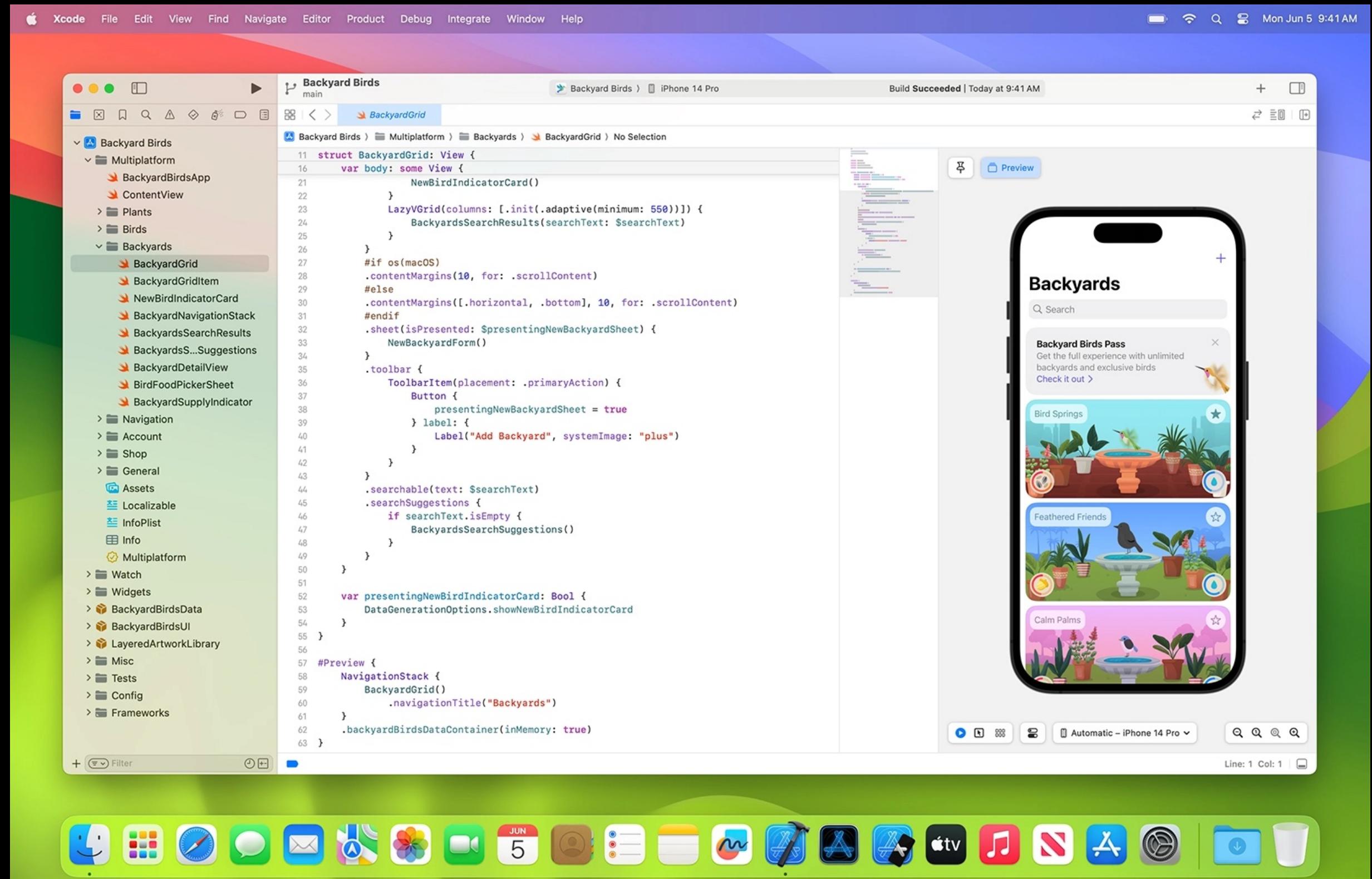
From Xcode to Neovim oder wie ich lernte, die Kommandozeile zu lieben



# From Xcode to Neovim

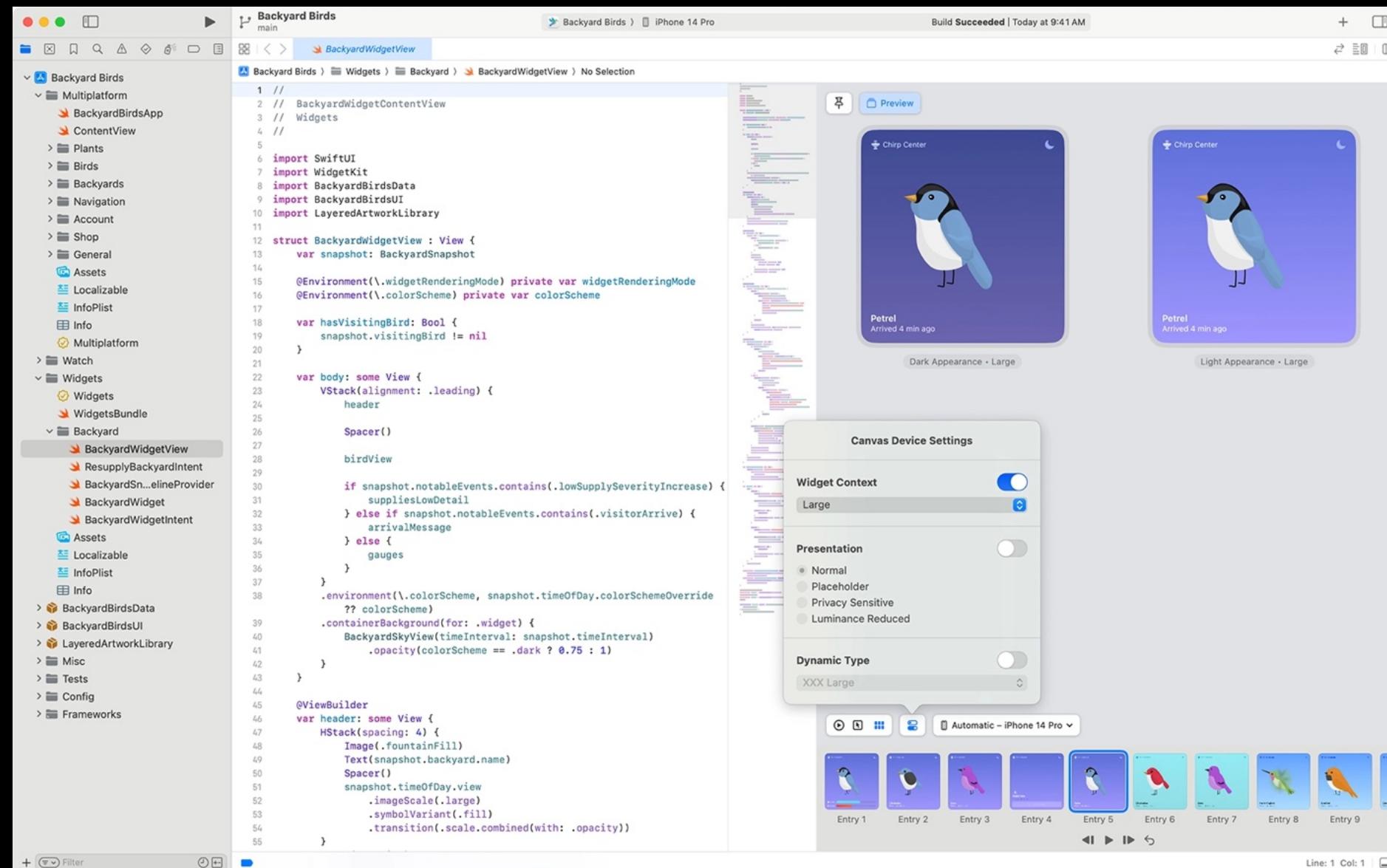
- » Xcode 
- » Blick über den Tellerrand 
- » Vim 
- » Mein Weg in die Freiheit 
- » Demo 

# Xcode



# The good

# SwiftUI Previews



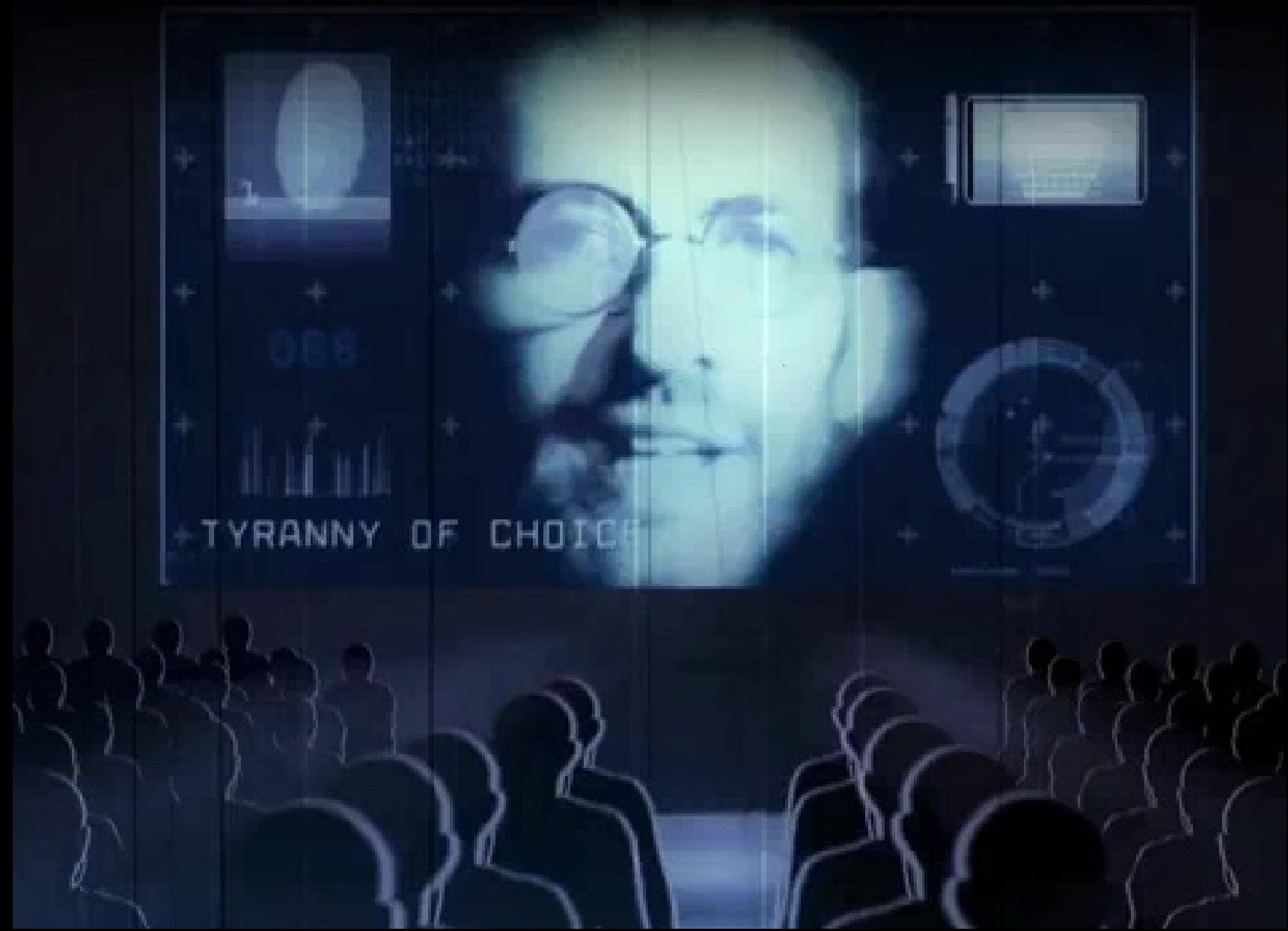
# Gesamtpaket aus einer Hand

- » Editor
- » UI Designer (IB oder SwiftUI)
- » Debugger
- » Device-Management
- » AppStore Anbindung

# 1st Party Open Source Tooling

- » Swift
- » Swift Package Manager
- » Swift Testing
- » Swift Build
- » SourceKit
- » Swift Syntax

# The bad



Apple entscheidet, womit wir wie entwickeln

- » keine wirkliche Unterstützung für andere Sprachen oder Plugins
- » aufgeblähte IDE (3D Editor, Interface Builder, etc.)
- » AI code assistance 
- » rudimentärer Editor
- » Refactoring? 
- » kaum Anbindung an 3rd-party-tooling

# The ugly



@m\_mlr@mastodon.social (iOS/mac/Android at KF Interactive)

- » nicht anpassbar an eigene Bedürfnisse
- » Abstürze beim Wechseln zwischen git branches
- » Derived Data Wüste (zig GB)
- » keine Unterstützung für grosse Dateien (json, logs, etc.)
- » Versionskontrolle 
- » [...] your favorite rant here



*...die Ruhe vor dem Sturm (23 Uhr)...*



# The others

The screenshot shows an Android Studio interface with the following components:

- Left Panel:** Shows the project structure with a file named `NiaApp.kt` open.
- Code Editor:** Displays Kotlin code for a preview screen. The code uses Jetpack Compose to build a UI with a top bar, content insets, and a `ForYouScreen` component.
- Emulator:** Two instances of the Android emulator are visible, both showing a placeholder screen for "Android Basics with Compose".
- Gemini AI Interface:** A sidebar titled "Gemini" contains a large text area with the heading "Hello, Android Developer" and "How can I help?". It also lists several example prompts and a "How do I fix my crash?" section.
- Bottom Panel:** Shows the Firebase Crashlytics dashboard with a list of issues and device statistics.

The screenshot shows a Microsoft Edge browser window displaying the source code for `button.ts` from `vscode.dev`. The browser interface includes a header with tabs for `my-app — button.ts` and `vscode.dev`, a search bar, and a toolbar with various icons. The left sidebar is the Explorer view from VS Code, showing a tree structure of files under `MY-APP`, with `button.ts` selected. The main content area shows the TypeScript code for the `Button` class:

```
15  export class Button extends Disposable implements IButton {  
16    protected options: IButtonOptions;  
17    protected _element: HTMLElement;  
18    protected _label: string | IMarkdownString = '';  
19  
20    private _onDidClick = this._register(new Emitter<Event>());  
21    get onDidClick(): BaseEvent<Event> {  
22      return this._onDidClick.event;  
23    }  
24  
25    private focusTracker: IFocusTracker;  
26  
27    constructor(container: HTMLElement, options: IButtonOptions) {  
28      super();  
29  
30      this.options = options;  
31  
32      this._element = document.createElement('a');  
33      this._element.classList.add('monaco-button');  
34      this._element.tabIndex = 0;  
35      this._element.setAttribute('role', 'button');
```

# Open Source

We think it's great !

emacs@blueberry

File Edit Options Buffers Tools Emacs-Lisp Help

Save Undo

```
(defsubst hash-table-empty-p (hash-table)
  "Check whether HASH-TABLE is empty (has 0 elements)."
  (zerop (hash-table-count hash-table)))

(defsubst hash-table-keys (hash-table)
  "Return a list of keys in HASH-TABLE."
  (let ((keys '()))
    (maphash (lambda (k _v) (push k keys)) hash-table)
    keys))

(defsubst hash-table-values (hash-table)
  "Return a list of values in HASH-TABLE."
  (let ((values '()))
    (maphash (lambda (k _v) (push v values)) hash-table)
    values))

-:--- subr-x.el.gz 36% L148 (Emacs-Lisp)
```

Next: [Distrib](#), Up: [\(dir\)](#)  
[\(emacs\)Top](#)

## The Emacs Editor

Emacs is the extensible, customizable, self-documenting real-time display editor. This manual describes how to edit with Emacs and some of the ways to customize it; it corresponds to GNU Emacs version 26.0.50.

If you are reading this in Emacs, type 'h' to read a basic introduction to the Info documentation system.

U:%%- \*info\* (emacs) Top Top L9 (Info Narrow)

```
 paul@ubuntu: ~
VIM - Vi IMproved
version 8.2.3741
by Bram Moolenaar et al.
Modified by team+vim@tracker.debian.org
Vim is open source and freely distributable

Sponsor Vim development!
type :help sponsor<Enter>    for information

type :q<Enter>          to exit
type :help<Enter> or <F1> for on-line help
type :help version8<Enter> for version info

0,0-1      All
```



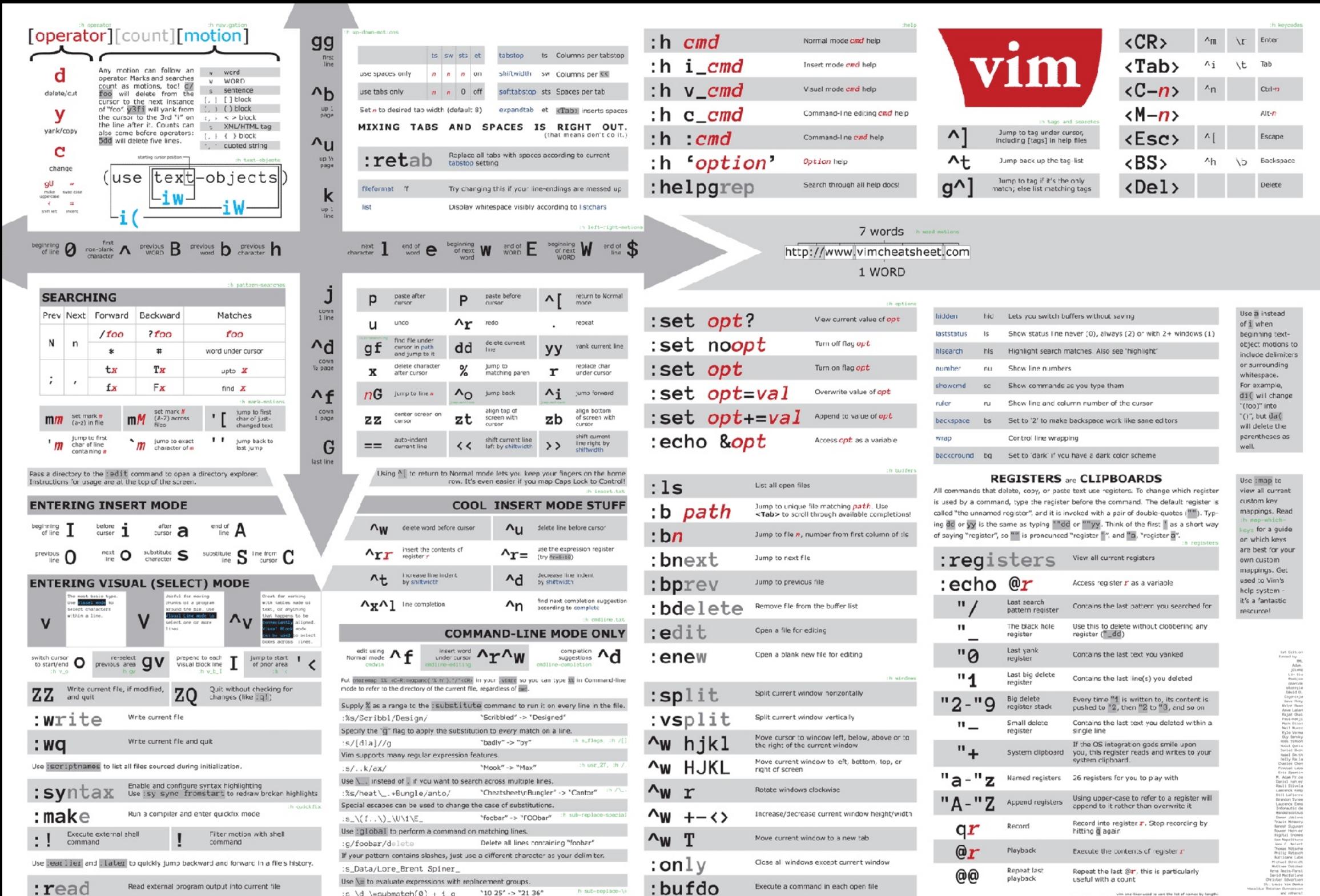
# #lifegoals



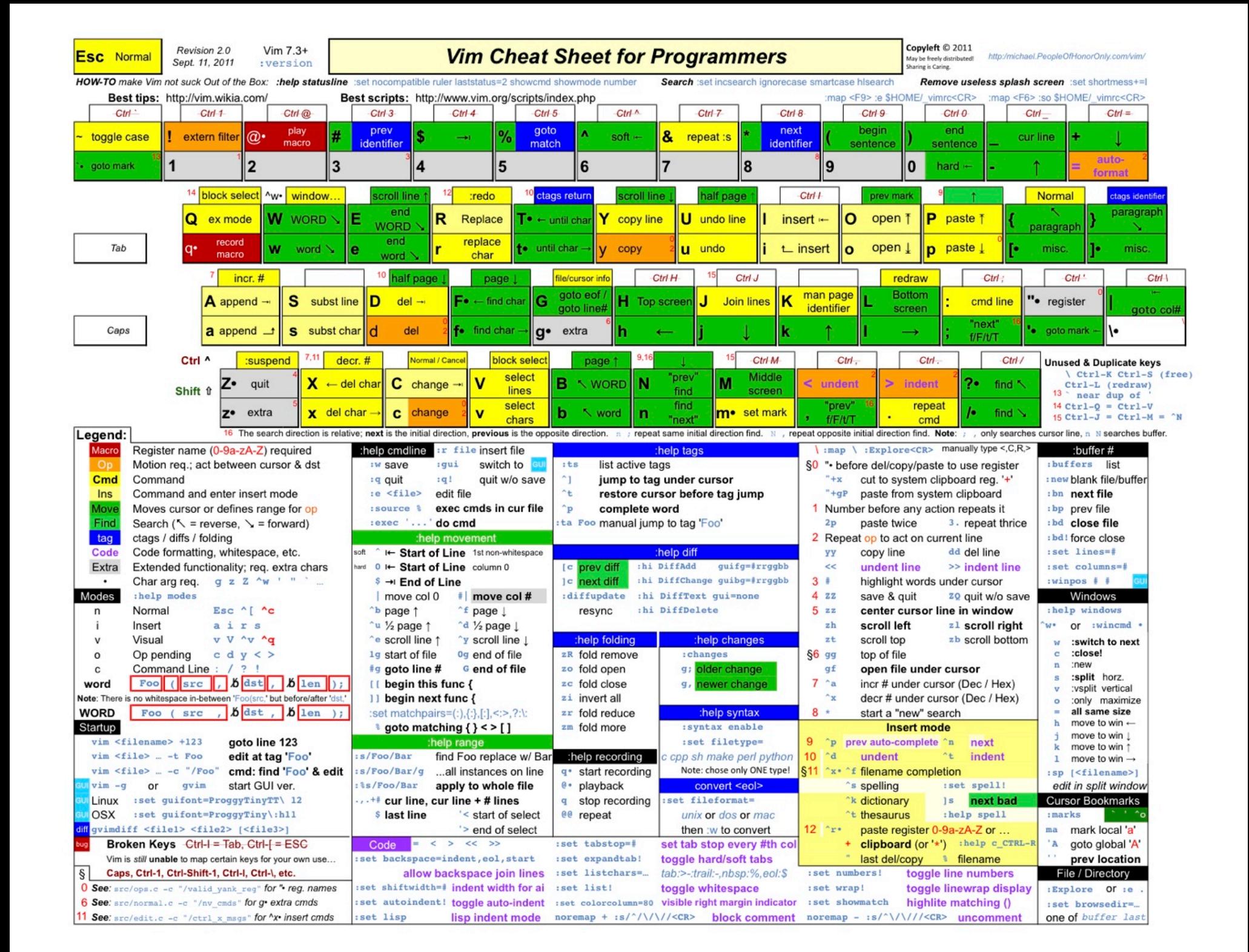
@m\_mlr@mastodon.social (iOS/mac/Android at KF Interactive)

# Learning Vim

## #lifegoals



@m\_mlr@mastodon.social (iOS/mac/Android at KF Interactive)

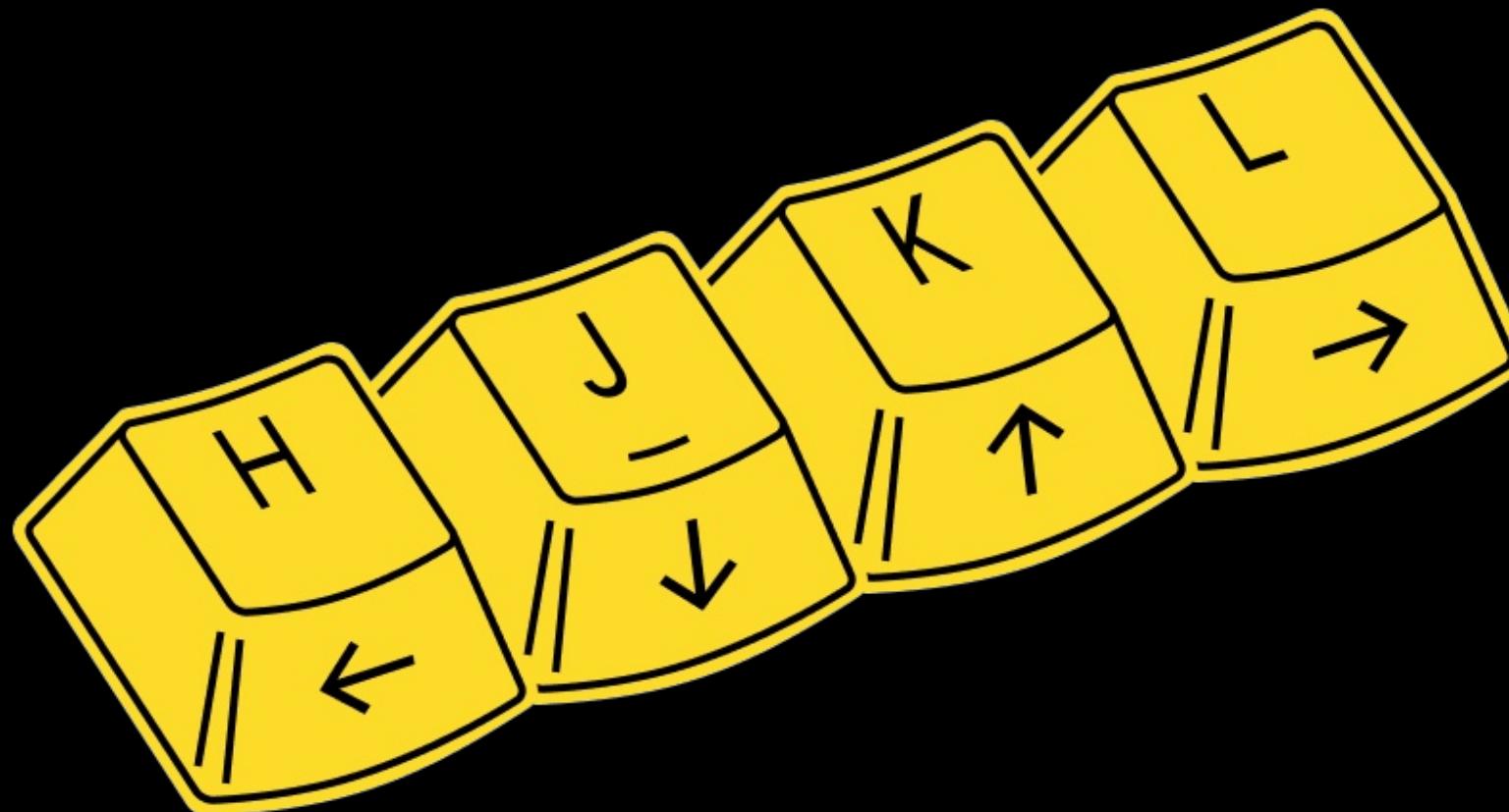


Danke!

Fragen?

...just kidding 😊

# Modal editing



# Vi

» irgendwann in den 70ern von Bill Joy entwickelt

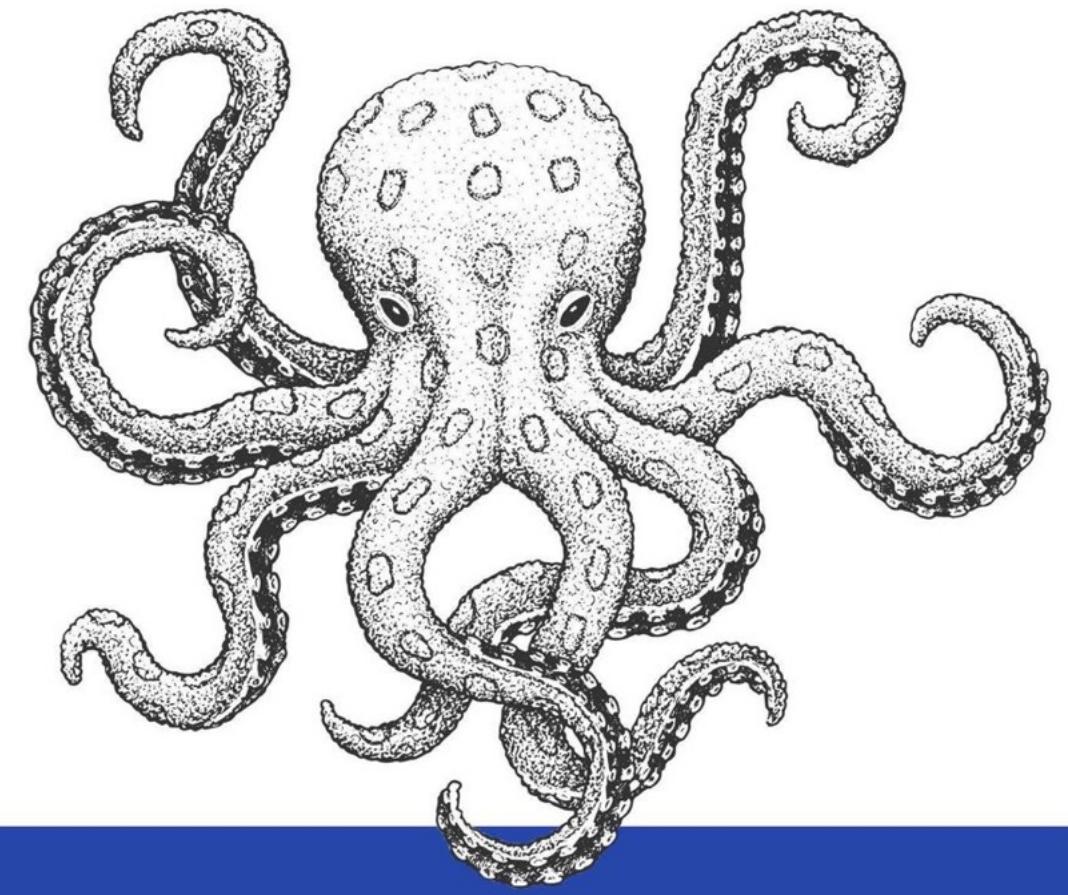


# Vim

- » von Bram Moolenaar Ende der 80er auf dem Amiga entwickelt
- » 1991 veröffentlicht



*Just memorize these fourteen contextually dependant instructions*



# Exiting Vim

*Eventually*

O RLY?

@ThePracticalDev

# Vim motions

# Cursor movement

**h** - move cursor left

**j** - move cursor down

**k** - move cursor up

**l** - move cursor right

**w** - jump forwards to the start of  
a word

**e** - jump forwards to the end of a  
word

**b** - jump backwards to the start  
of a word

**ge** - jump backwards to the end  
of a word

# Extended motions

**2w** - jump forwards 2 words

**10j** - jump down 10 lines

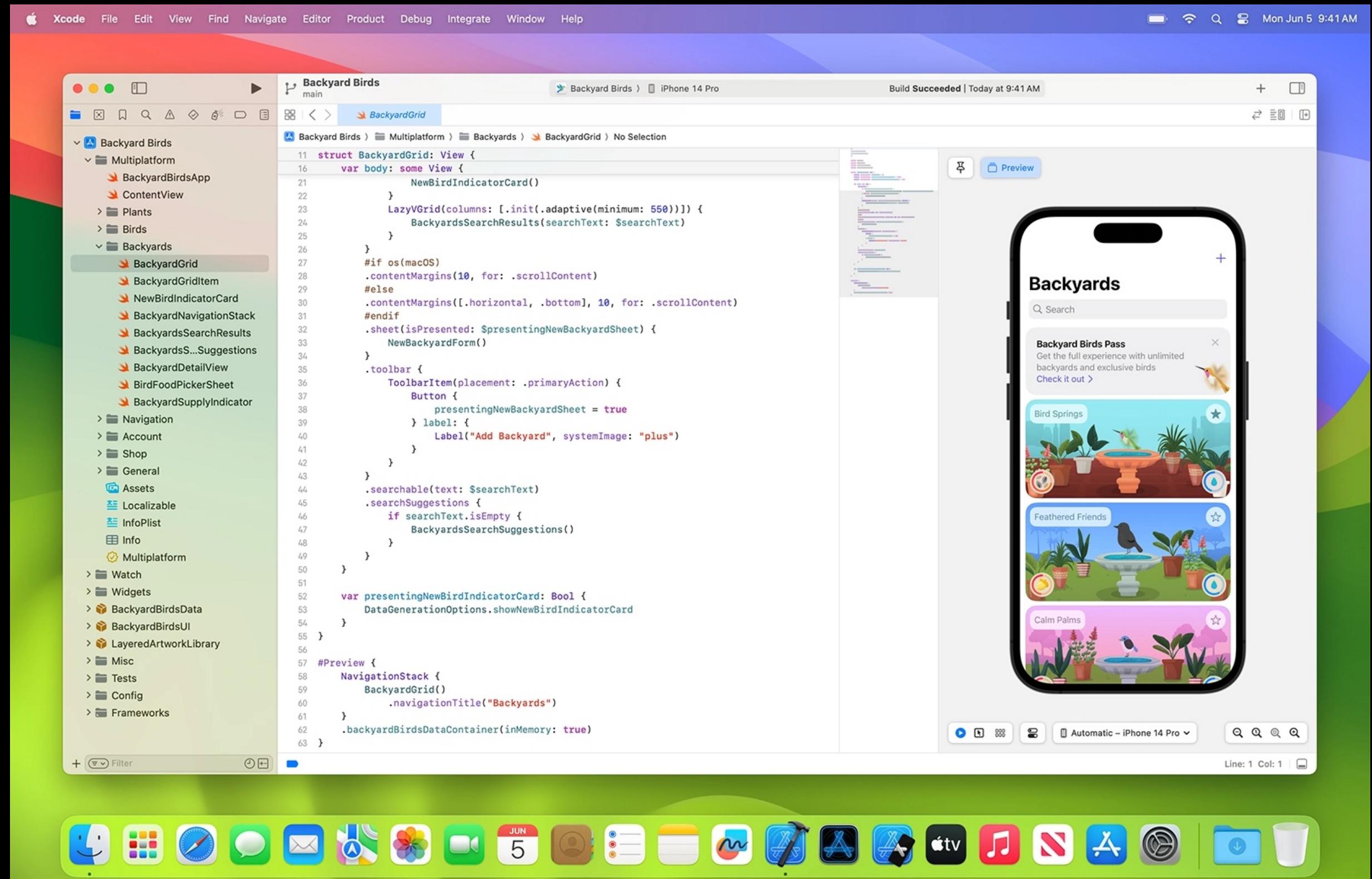
**y5w** - copy 5 words

**y3k** - copy 3 lines up

**d7j** - delete 7 lines down

# Demo

# Das Problem



# Die Lösung



@m\_mlr@mastodon.social (iOS/mac/Android at KF Interactive)

```
xcodetool -list -json  
xcodetool -scheme MyApp -showdestinations  
xcodetool -scheme MyApp -destination platform=macos,id=...,arch=arm64e build  
xcodetool -scheme MyApp -destination platform=macos,id=...,arch=arm64e \  
-showBuildSettings -json
```

```
xcrun simctl boot "iPhone 13"
open Simulator.app
xcrun simctl install "iPhone 13" MyApp.app
xcrun simctl launch "iPhone 13" /some/obscure/path/to/my/build/app
```

```
swift test --enable-swift-testing -c debug --xunit-output junit.xml
```

lldb anyone?

## How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the fucking owl



# neovim

- » hyperextensible Vim-based text editor
- » Vim fork, entwickelt seit 2014

# Mission statement

- » Simplify maintenance and encourage contributions
- » Split the work between multiple developers
- » Enable advanced UIs without modifications to the core
- » Maximize extensibility

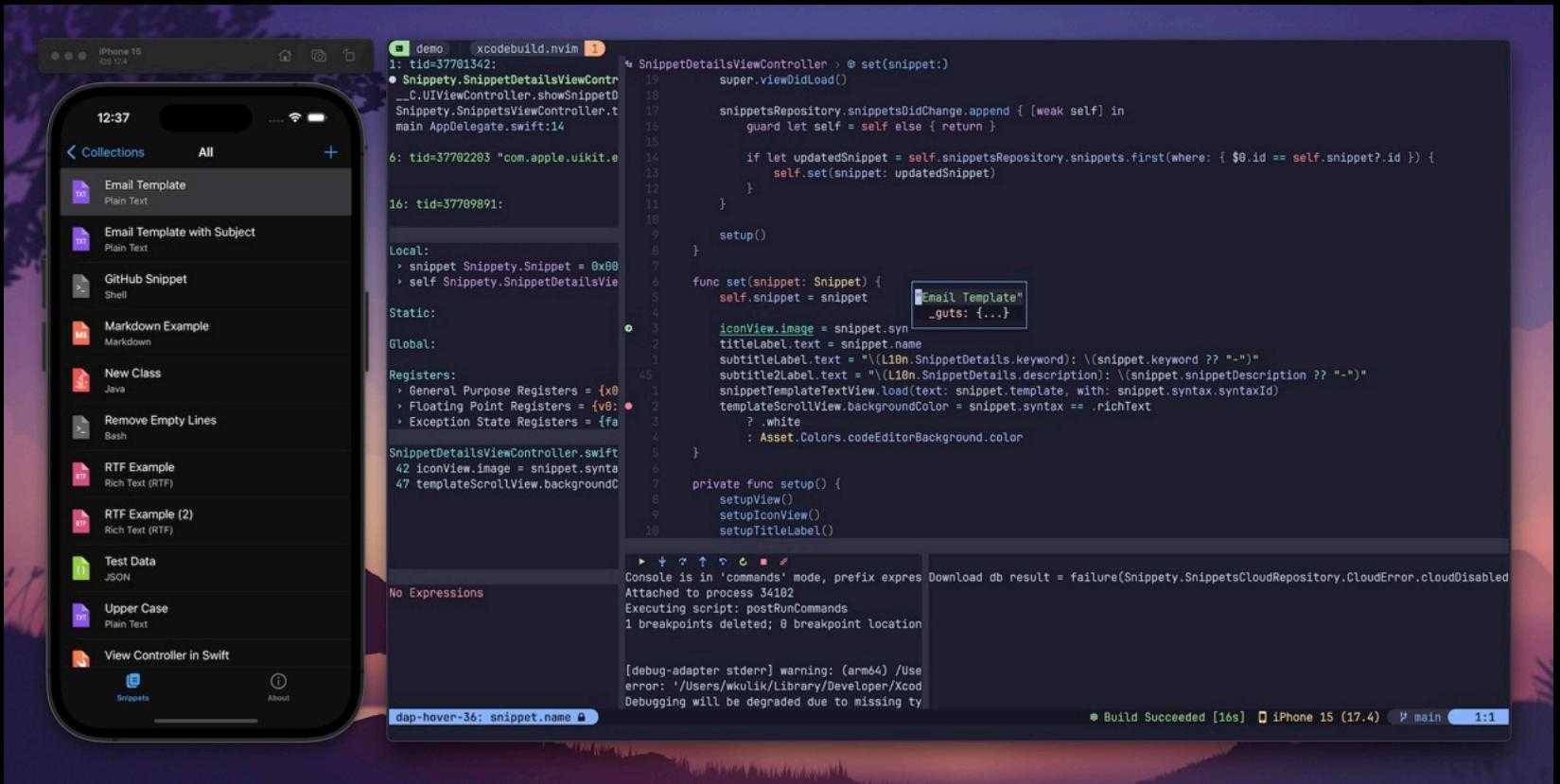


# LSP



Danke, Bill!

# xcodebuild.nvim von Wojciech Kulik 🙌

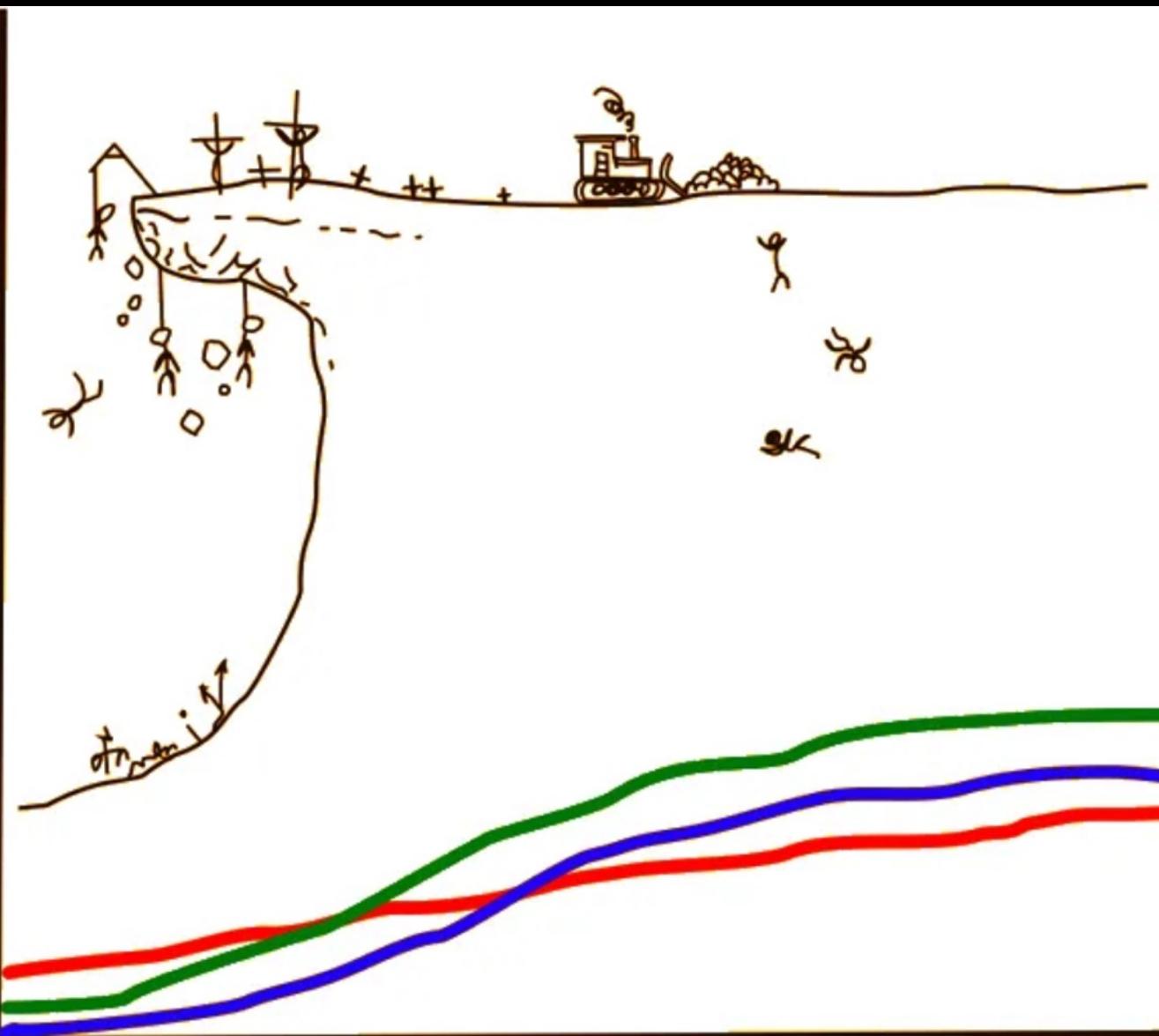


Why Switching From Xcode to Neovim Can Become The Best Decision You Ever Made

# Getting out of my comfort zone



😱 Don't be scared! 😱



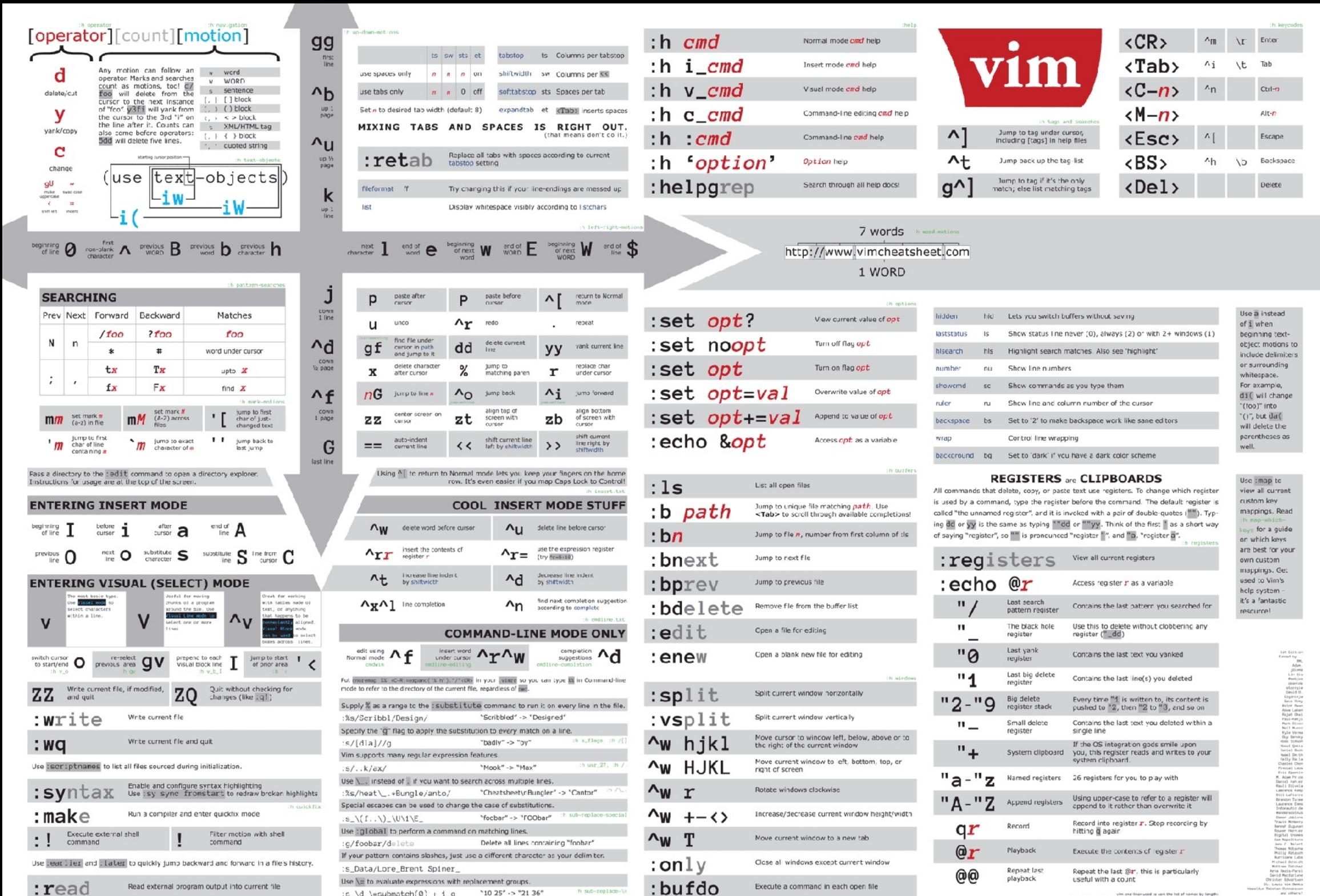
# LazyVim



Avoiding the config 🐰 hole

*:h*

*there is always help available*



@m\_mlr@mastodon.social (iOS/mac/Android at KF Interactive)

...some weeks later...

Demo

# PDE (personal development environment)

- » Freiheit
- » Flexibilität dank Lua und riesigen Plugin-Ökosystem
- » *Better understanding of your tool belt™*
- » CLI hat ihren Schrecken verloren

# PDE (personal development environment)

- » Entdeckung der wunderbaren Welt der Kommandozeile
- » **viel** neues gelernt (Vim, Lua, 10 Finger System) 
- » deprecation of the 

# Danke

# Links

- » [Neovim](#)
- » [LazyVim](#)
- » [NeoVim for the ambitious developer](#)
- » [xcodebuild.nvim](#)
- » [Neoxcd](#)
- » [Neotest Swift Testing](#)
- » [The Primeagen](#)
- » [TJ DeVries](#)

@m\_mlr@mastodon.social (iOS/mac/Android at KF Interactive)