

Bun もすなる Zig といふものを
してみむとてするなり

自己紹介

Self Introduction

2025.8.26

自己紹介

設楽 朗人 Akito Shitara

京都大学大学院 情報学研究科 修士1年。

機械学習を用いた時系列予測に関する研究に従事。

IT×教育の事業を行う会社で、大学生メンターとして中高生にプログラミングを教えている。

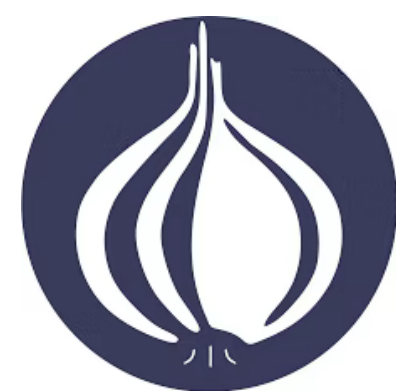
Web サービスの会社でお手伝い中。

一人暮らし歴5年目。

卵でとじる系の料理が好き。

方向音痴。

よく書く言語 Frequently used programming languages



Perl



TypeScript



Python



Rust



Zig とは？

What is Zig?

2025.8.26

Zig 使ってみた



いわゆる Better C

- Robust

メモリ不足などのエッジケースでも動作が正しく行われる。

- Optimal

プログラムの動作やパフォーマンスが最適になるように書く。

- Reusable

制約の異なる多くの環境で同じコードが動作する。

- Maintainable

コードを読むためのオーバーヘッドが少なく、要件や環境の変化に強い。

– クロスコンパイル可能

LLVM をバックエンドとして使っており、標準でクロスコンパイルをサポートしている。

– めちゃめちゃ速い

Rust, C などとほぼ同じ。場合によりこれらよりも速いらしい。

– C との互換性が高い

C ABI に乗っているライブラリは全て使える。

- まだ正式リリースではない

ので、破壊的変更が日々行われている

- 最近で言うと、`async / await` が復活するらしい
 - <https://kristoff.it/blog/zig-new-async-io/>

AI 君に実装をお願いすると、かなり辛い感じになる

- エコシステムが貧弱

固まり切っていない言語仕様の上にライブラリを作るのが難しい

✖ 大規模なコードベース

個人規模では Zig の破壊的変更についていきながら新機能を継続的に開発するのは効率的ではない

✖ 外部ライブラリに依存したいサービス

タイムゾーンの処理や SDK などがほぼ存在しない世界観なので現状自前で実装するしかない

そんな...

2025.8.26

閑話休題

閑話休題

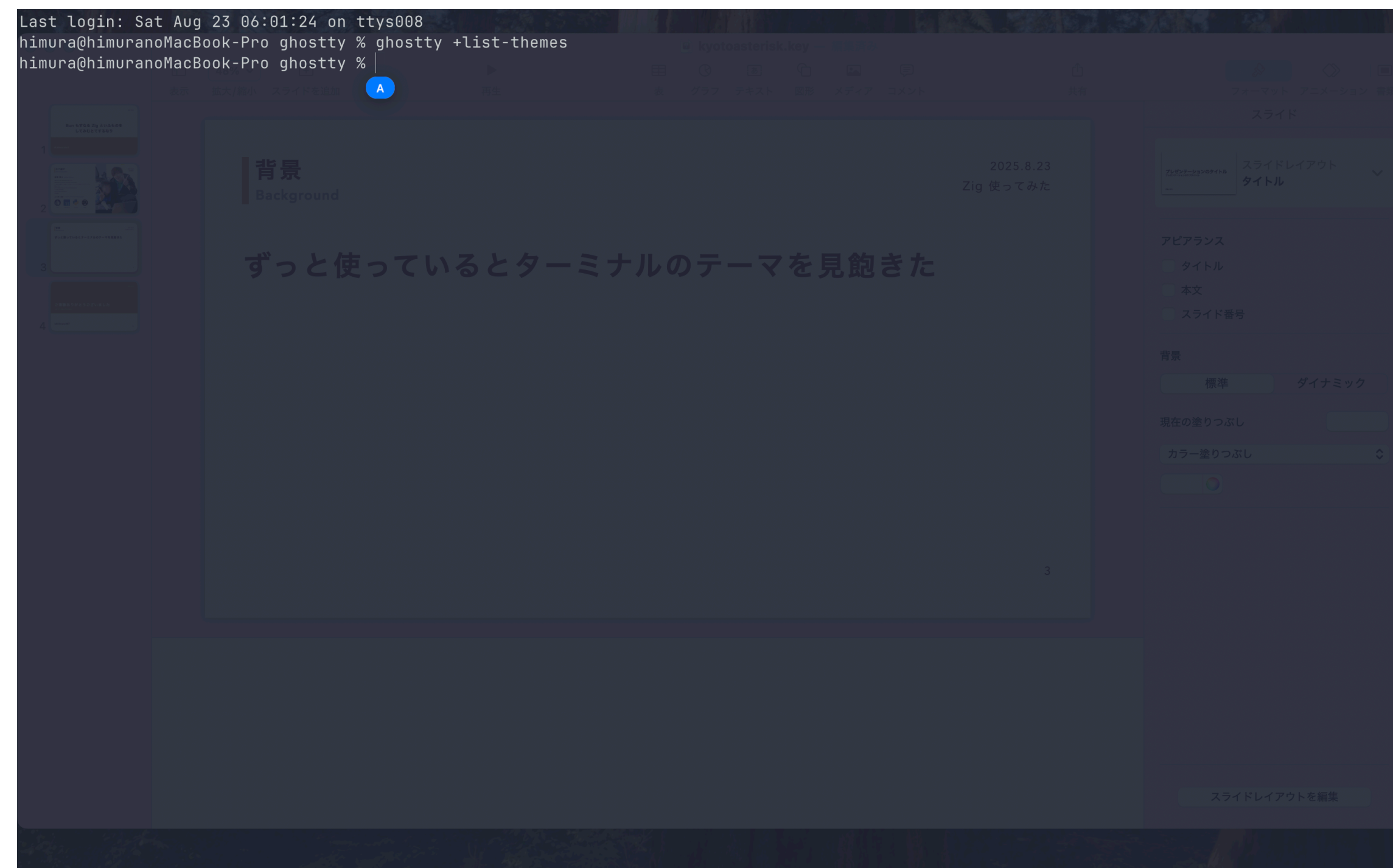
最近の悩み

Background

2025.8.26

閑話休題

ターミナルのテーマを見飽きた



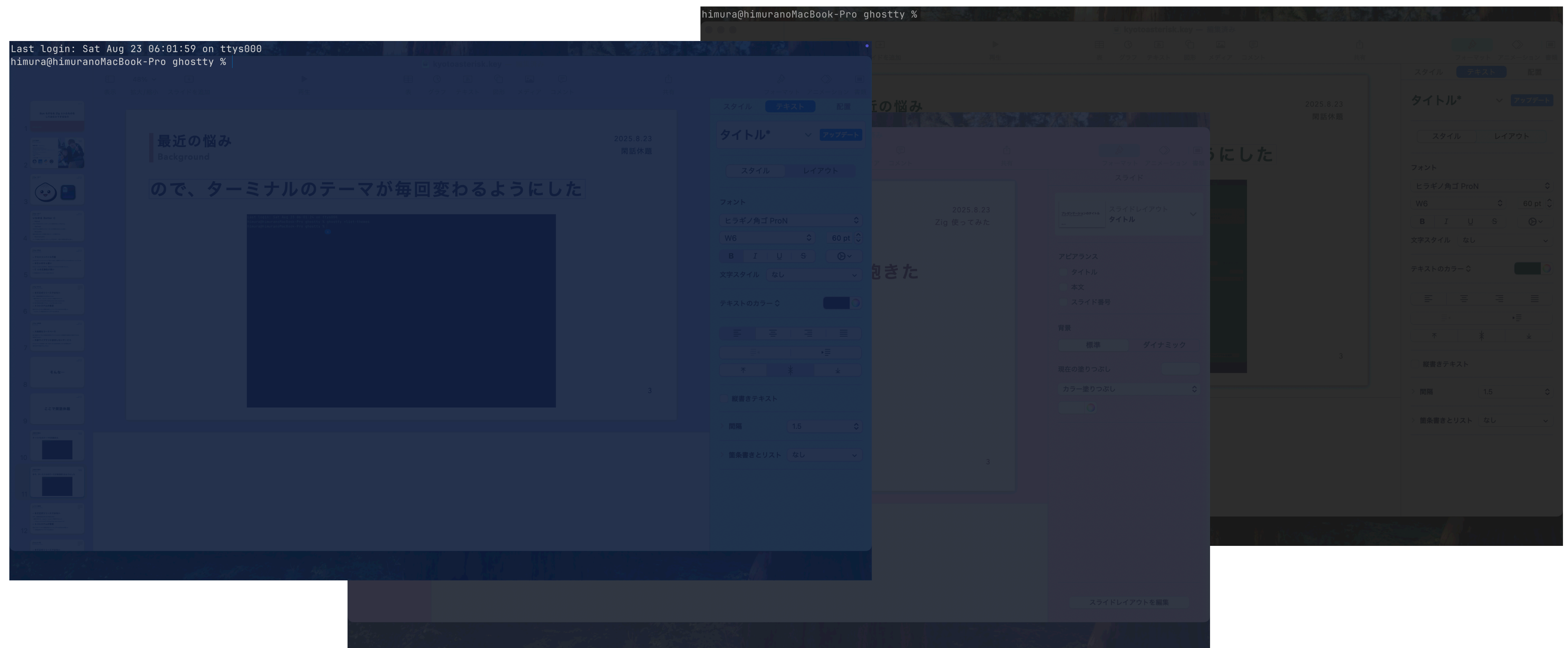
最近の悩み

Background

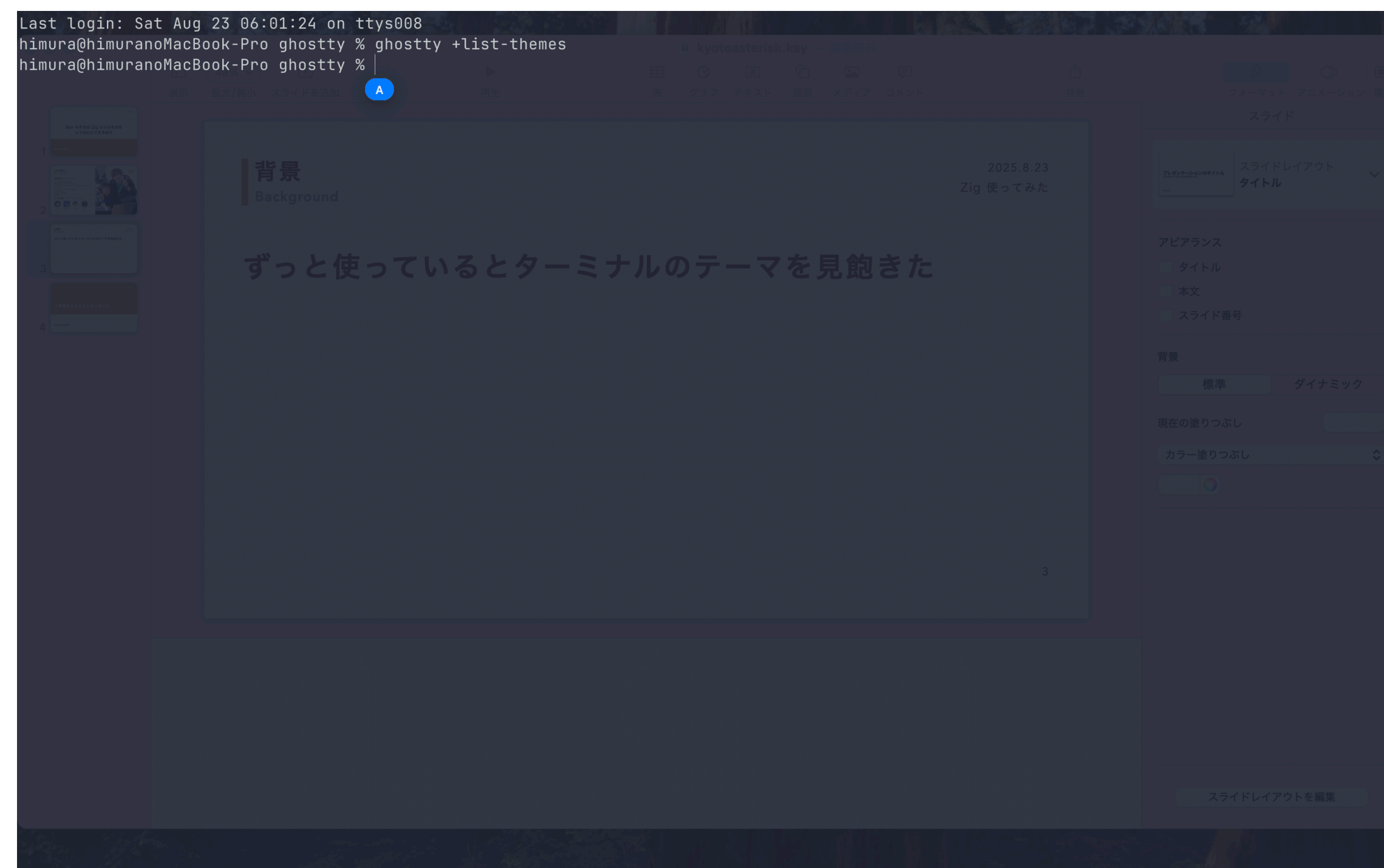
2025.8.26

閑話休題

ので、ターミナルのテーマが毎回変わるようにした



また、Cmd+Shift+f でお気に入りテーマを 保存できるようにした



(ランダム選択時に) たまに”強い”背景が選択される

synthwave-everything

SynthwaveAlpha

Tango Adapted

Tango Half Adapted

Teerb

terafox

Terminal Basic

Thayer Bright

The Hulk

Tinacious Design (Dark)

Tinacious Design (Light)

tokyonight

tokyonight-day

tokyonight-storm

tokyonight_moon

tokyonight_night

Tomorrow

Tomorrow Night

Tomorrow Night Blue

Tomorrow Night Bright

Tomorrow Night Burns

Tomorrow Night Eighties

ToyChest

Treehouse

Twilight

Ubuntu

UltraDark

UltraViolent

UnderTheSea

Unikitty

Urple

Vaughn

vesper

VibrantInk

vimbones

Violet Dark

Violet Light

WarmNeon

Wez

Whimsy

WildCherry

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

bat ziggzagg.zig

File: ziggzagg.zig

1const std = @import("std");

2

3pub fn main() !void {

4const stdout = std.io.getStdOut().writer();

5var i: usize = 1;

6while (i ≤ 16) : (i += 1) {

7if (i % 15 == 0) {

8try stdout.writeAll("ZiggZagg\n");

9} else if (i % 3 == 0) {

10try stdout.writeAll("Zigg\n");

11} else if (i % 5 == 0) {

12try stdout.writeAll("Zagg\n");

13} else {

14try stdout.print("{d}\n", .{i});

15}

16}

17}

ghostty on 主 main [+] via 0.13.0 via impure (ghostty-env)

+ at 10:36:15 →

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras hendrerit aliquet turpis non dictum. Mauris pulvinar nisl sit amet dui cursus tempus. Pellentesque ut dui justo. Etiam quis magna sagittis nisi pretium consequat vitae ut nisl. Sed at metus id odio pulvinar sodales. Vestibulum sollicitudin, sem id tristique vestibulum, neque ante dictum tortor, in convallis mi enim ac lorem. Suspendisse orci ex, ullamcorper sed leo vitae, mattis egestas nisl. Morbi id est vel ipsum mollis convallis vel at mauris. Duis vehicula facilisis placerat. Aliquam venenatis auctor lorem vel elementum. Proin ac tincidunt lacus. Sed facilisis tellus

Galaxy

Galizur

Ghostty Default StyleDark

Github

Github Dark

Github-Dark-Colorblind

Github-Dark-Default

Github-Dark-Dimmed

Github-Dark-High-Contrast

Github-Light-Colorblind

Github-Light-Default

Github-Light-High-Contrast

GitLab-Dark

GitLab-Dark-Grey

GitLab-Light

Glacier

Grape

Grass

Grey-green

gruber-darker

gruvbox-material

GruvboxDark

GruvboxDarkHard

GruvboxLight

GruvboxLightHard

Guezwhoz

Hacktober

Hardcore

Harper

Havn Daggrgy

Havn Skumring

HaXOR_BLUE

HaXOR_GR33N

heeler

Highway

Hipster Green

Hivacruz

Homebrew

Hopscotch

Hopscotch.256

HaXOR_R3D

/Applications/Ghostty.app/Contents/Resources/ghostty/themes/HaXOR_R3D

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

bat ziggzagg.zig

File: ziggzagg.zig

1const std = @import("std");

2

3pub fn main() !void {

4const stdout = std.io.getStdOut().writer();

5var i: usize = 1;

6while (i ≤ 16) : (i += 1) {

7if (i % 15 == 0) {

8try stdout.writeAll("ZiggZagg\n");

9} else if (i % 3 == 0) {

10try stdout.writeAll("Zigg\n");

11} else if (i % 5 == 0) {

12try stdout.writeAll("Zagg\n");

13} else {

14try stdout.print("{d}\n", .{i});

15}

16}

17}

ghostty on 主 main [+] via 0.13.0 via impure (ghostty-env)

+ at 10:36:15 →

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras hendrerit aliquet turpis non dictum. Mauris pulvinar nisl sit amet dui cursus tempus. Pellentesque ut dui justo. Etiam quis magna sagittis nisi pretium consequat vitae ut nisl. Sed at metus id odio pulvinar sodales. Vestibulum sollicitudin, sem id tristique vestibulum, neque ante dictum tortor, in convallis mi enim ac lorem. Suspendisse orci ex, ullamcorper sed leo vitae, mattis egestas nisl. Morbi id est vel ipsum mollis convallis vel at mauris. Duis vehicula facilisis placerat. Aliquam venenatis auctor ipsum vel elementum. Proin ac tincidunt lacus. Sed facilisis tellus

13

処理は Perl で書いているが、以下の要件を満たせば
どんな言語でもいい

- クロスコンパイル可能
- Random 処理の seed の固定ができる
- ファイルの `theme =` から始まる行を書き換える
 - Regex をサポートしていると嬉しい

処理は Perl で書いているが、以下の要件を満たせば

どんな言語でもいい

– クロスコンパイル

– Random 処理の固定ができる

– フォーマット指定で、` から始まる行を書き換える

– Random 処理をしていると嬉しい

くできらあ！



Ziggy くん

処理は Perl で書いているが、以下の要件を満たせば

どんな言語でもいい

– クロスコンパイル

– Random 処理の固定ができる

– フォーマット指定で、` から始まる行を書き換える

– Random 処理をしていると嬉しい



くできらあ！

ターミナル背景色の推薦システムを Zig で実装する

Terminal background recommendation system

Zig のテーマ

Theme location

2025.8.26

Zig のテーマ

The second directory is the `themes` subdirectory of the Ghostty resources directory. Ghostty ships with a multitude of themes that will be installed into this directory. On macOS, this list is in the `Ghostty.app/Contents/Resources/ghostty/themes` directory. On Linux, this list is in the `share/ghostty/themes` directory (wherever you installed the Ghostty "share" directory).

```
palette = 0=#51576d
palette = 1=#e78284
palette = 2=#a6d189
palette = 3=#e5c890
palette = 4=#8caae6
palette = 5=#f4b8e4
palette = 6=#81c8be
palette = 7=#a5adce
palette = 8=#626880
palette = 9=#e67172
palette = 10=#8ec772
palette = 11=#d9ba73
palette = 12=#7b9ef0
palette = 13=#f2a4db
palette = 14=#5abfb5
palette = 15=#b5bfe2
background = #303446
foreground = #c6d0f5
cursor-color = #f2d5cf
cursor-text = #c6d0f5
selection-background = #626880
selection-foreground = #c6d0f5
```

テーマの解析

Theme Analysis

2025.8.26

テーマの解析

■ 各種テーマからカラーコードをパース

#RRGGBB の形に直してから構造体の形にする

■ パースしたカラーコードを HSL 色空間上にマッピング

→のような形式で RGM の値を HSL 色空間上の座標に変換する

H: 色相

S: 彩度

L: 輝度

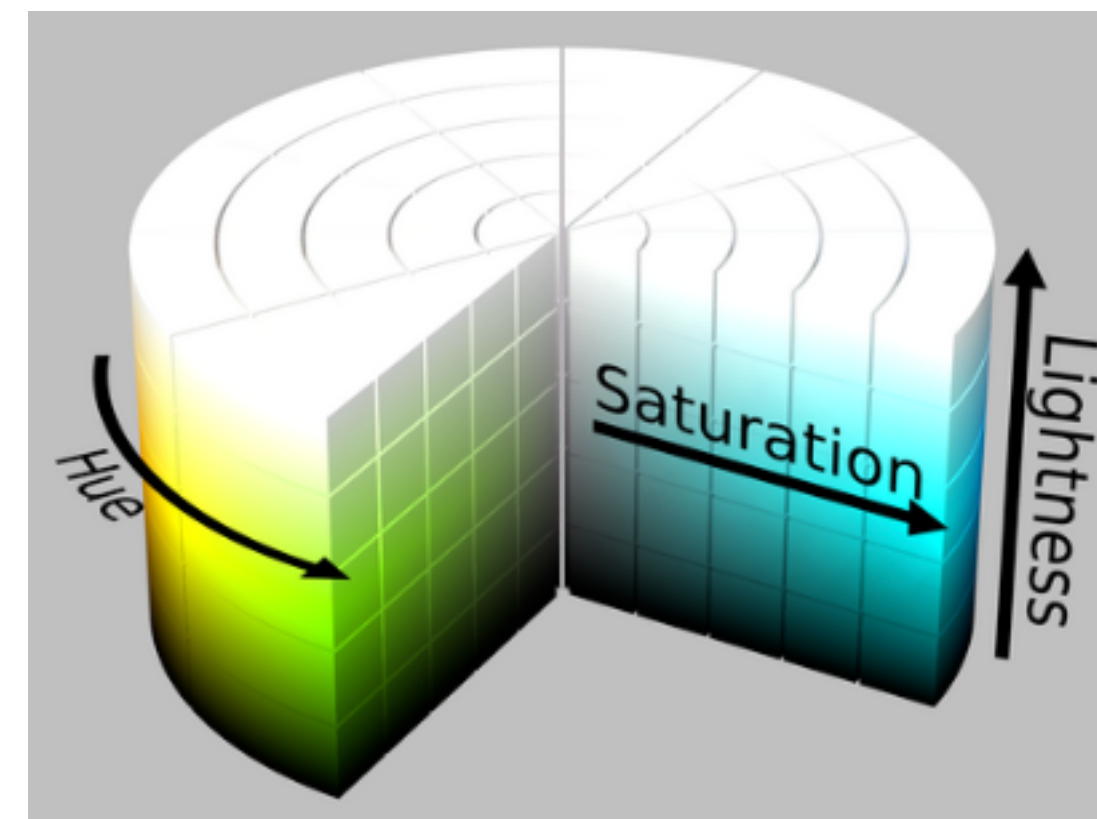
■ 距離的に近いテーマを選択して推薦

ユーザーが設定したお気に入りのテーマを基に

- background

- foreground

の色に重みをつけて距離的に近いテーマを検索して推薦する



```
const RGB: type = struct {  
    r: f32,  
    g: f32,  
    b: f32,  
};  
  
const HSL: type = struct {  
    h: f32, // 0-360  
    s: f32, // 0-1  
    l: f32, // 0-1  
};
```


ターミナルのテーマの更新

Theme Update

2025.8.26

テーマの解析

- テーマの更新ロジックを以下のように変更

Before

- 50% の確率でランダムにテーマを選択
- 50% の確率でお気に入りテーマの中から選択

After

- 20% の確率でランダムにテーマを選択
- 30% の確率でお気に入りテーマの中から選択
- 50% の確率で推薦テーマ上位 10 件の中からランダムに選択

- 実行速度

速いのが売りだが、そもそも元のスクリプトの実行時間が 20ms とかだったので
大して変わらなかった

- 多少は速くなった

【おまけ】

Zig を使っていて嬉しかったこと

What made me happy about using Zig

Allocator

```
fn getAvailableThemes(allocator: Allocator) !ArrayList([]const u8) {
    const result = try executeCommand(allocator, &.{ "ghostty", "+list-themes" });

    var themes = ArrayList([]const u8){};
    var lines = std.mem.splitSequence(u8, result.stdout, "\n");

    while (lines.next()) |line| {
        if (line.len == 0) continue;

        // Remove " (builtin)" or " (user)" suffix if present
        var theme_name = line;
        if (std.mem.indexOf(u8, line, " (")) |index| {
            theme_name = line[0..index];
        }

        try themes.append(allocator, try allocator.dupe(u8, theme_name));
    }

    return themes;
}
```

何が嬉しいのか

Nani ga ureshii no?

2025.8.26

メモリ管理

単体テストが 簡単にできる

```
fn getAvailableThemes(allocator: Allocator) !ArrayList([]const u8) {
    const result = try executeCommand(allocator, &.{ "ghostty", "+list-themes" });

    var themes = ArrayList([]const u8){};
    var lines = std.mem.splitSequence(u8, result.stdout, "\n");

    while (lines.next()) |line| {
        if (line.len == 0) continue;

        // Remove " (builtin)" or " (user)" suffix if present
        var theme_name = line;
        if (std.mem.indexOf(u8, line, " (")) |index| {
            theme_name = line[0..index];
        }

        try themes.append(allocator, try allocator.dupe(u8, theme_name));
    }

    return themes;
}
```

パッと見 free してないのでメモリリーク起こしそう

Free wasureteisou

2025.8.26

メモリ管理



```
fn getAvailableThemes(allocator: Allocator) !ArrayList([]const u8) {
    const result = try executeCommand(allocator, &.{ "ghostty", "+list-themes" });

    var themes = ArrayList([]const u8){};
    var lines = std.mem.splitSequence(u8, result.stdout, "\n");

    while (lines.next()) |line| {
        if (line.len == 0) continue;

        // Remove " (builtin)" or " (user)" suffix if present
        var theme_name = line;
        if (std.mem.indexOf(u8, line, " (")) |index| {
            theme_name = line[0..index];
        }

        try themes.append(allocator, try allocator.dupe(u8, theme_name));
    }

    return themes;
}
```


パッと見 free してないのでメモリリーク起こしそう

Free wasureteisou

2025.8.26

メモリ管理

ArenaAllocator

CLI アプリケーションなど開始から終了まで一直線に終わる処理には

ArenaAllocator という Allocatorを使う

確保したメモリを終了時にまとめて free してくれる

Run | Debug

```
pub fn main() !void {  
    var arena = std.heap.ArenaAllocator.init(std.heap.page_allocator);  
    defer arena.deinit();  
    const allocator = arena.allocator();
```


Result 型っぽいやつ

Error Union Type

2025.8.26

Error Union Type

特定のエラー (FileNotFound) は空のリストを返し
他のエラーは上位の処理に投げつける

```
fn getFavorites(allocator: Allocator, favorites_file: []const u8) !ArrayList([]const u8) {  
    const file = std.fs.cwd().openFile(favorites_file, .{}) catch |err| switch (err) {  
        error.FileNotFound => return ArrayList([]const u8){},  
        else => return err,  
    };  
    defer file.close();
```

defer

- あとでやることをスタックに積んでいく

errdefer

- エラーの時のみやることをスタックに積んでいく
 - 通常時はメモリ解放しなくてもいい場合とか

Zero Configuration Philosophy

Ghostty is designed to work out of the box with no configuration for most users. Ghostty has sensible defaults, embeds a default font (JetBrains Mono), has built-in nerd fonts, and more. We are constantly challenging ourselves as a project to eliminate any *necessary configuration* to use Ghostty.

To that end, if you find yourself needing to configure something other than highly subjective things like the theme and you think it should be a default, please [open a discussion](#).

And, if you're used to feeling you need to configure your terminal, maybe give Ghostty a try without any configuration first. You might be surprised!

ご清聴ありがとうございました

次回はターミナルエミュレータ自作編で！

Akito Shitara