Writing Accessible Go

julia ferraioli
Open Source @ Google
@juliaferraioli

Google Open Source

Transcript:
http://bit.ly/accessible-go-transcript

Hi, I'm julia

- Writer of code for people who write open source code
- Gopher for a few years
- Frequently understatedly described as "stubborn" AKA "stubborn AF"

Vision-impaired off-and-on for the past two years

GIF by Jim Haseloff: http://bit.ly/2uCJYCj

Uhhh…

Intermittent failures are the WORST

I was/am (intermittently) disabled

“[Disability] is a complex phenomenon, reflecting the interaction between features of a person’s body and features of the society in which [they live].”

- World Health Organization

Think about your tool set

Demo

```
package main

import (
        "fmt"
)

type Vector []float64

func main() {
        a, b := Vector{1, 2, 3}, Vector{4, 5, 6}
        a, b, _ = swap(a, b)
        fmt.Printf("Swapped vectors a: %v, b: %v\n", a, b)

        c := Vector{7, 8, 9}
        sum, _ := add([]Vector{a, b, c}...)
        fmt.Printf("Sum of all vectors: %v\n", sum)

        multiplier := 3
        scaled := scale(sum, multiplier)
        fmt.Printf("Scaled up by %d the sum is: %v\n", multiplier, scaled)
}

// swap swaps two vectors, and will return an error if the dimensions are incom\
-uu-:**-F1  vector_do.go    Top L1     (Fundamental)----------------------------
Auto-saving...done
```

We make products accessible, but not the processes by which they are built

Accessible tooling is not enough

Some hard-earned lessons in writing accessible code

Group code blocks logically

```
a, b := Vector{1, 2, 3}, Vector{4, 5, 6}
a, b, _ = swap(a, b)


c := Vector{7, 8, 9}
total, _ := add([]Vector{a, b, c}...)
```

OK

Group code blocks logically

```
a, b, c := Vector{1, 2, 3}, Vector{4, 5,\
            6}, Vector{7, 8, 9}
a, b, _ = swap(a, b)
total, _ := add([]Vector{a, b, c}...)
```

NOT RECOMMENDED

Group code blocks logically

```
a, b := Vector{1, 2, 3}, Vector{4, 5, 6}
a, b, _ = swap(a, b)

c := Vector{7, 8, 9}
total, _ := add([]Vector{a, b, c}...)
```

- Keep your variables close to where they are used
- Same with interfaces, struct, type declaration

Keep names short

```
var a, b Vector
```

OK

```
var vectorA, vectorB Vector
```

NOT RECOMMENDED

- Fast to listen to
- Easier to navigate around
- Less effort to type

Make names meaningful

```
var total, scaled Vector
```

OK

```
var tVec, sVec Vector
```

NOT RECOMMENDED

- Meaningful names reduce cognitive load
- A light form of self-documenting code
- Reduces the amount of jumping around in the codebase

Use pronounceable names

```
var total Vector                    OK
func add(...)                       OK

var tVec Vector                     NOT RECOMMENDED
func addAllVecs(...)                NOT RECOMMENDED
```

- Screenreaders can read them
- Takes less time than pronouncing a string of letters

Use new lines intentionally

```
a, b := Vector{1, 2, 3}, Vector{4, 5, 6}
a, b, _ = swap(a, b)

c := Vector{7, 8, 9}
total, _ := add([]Vector{a, b, c}...)
```

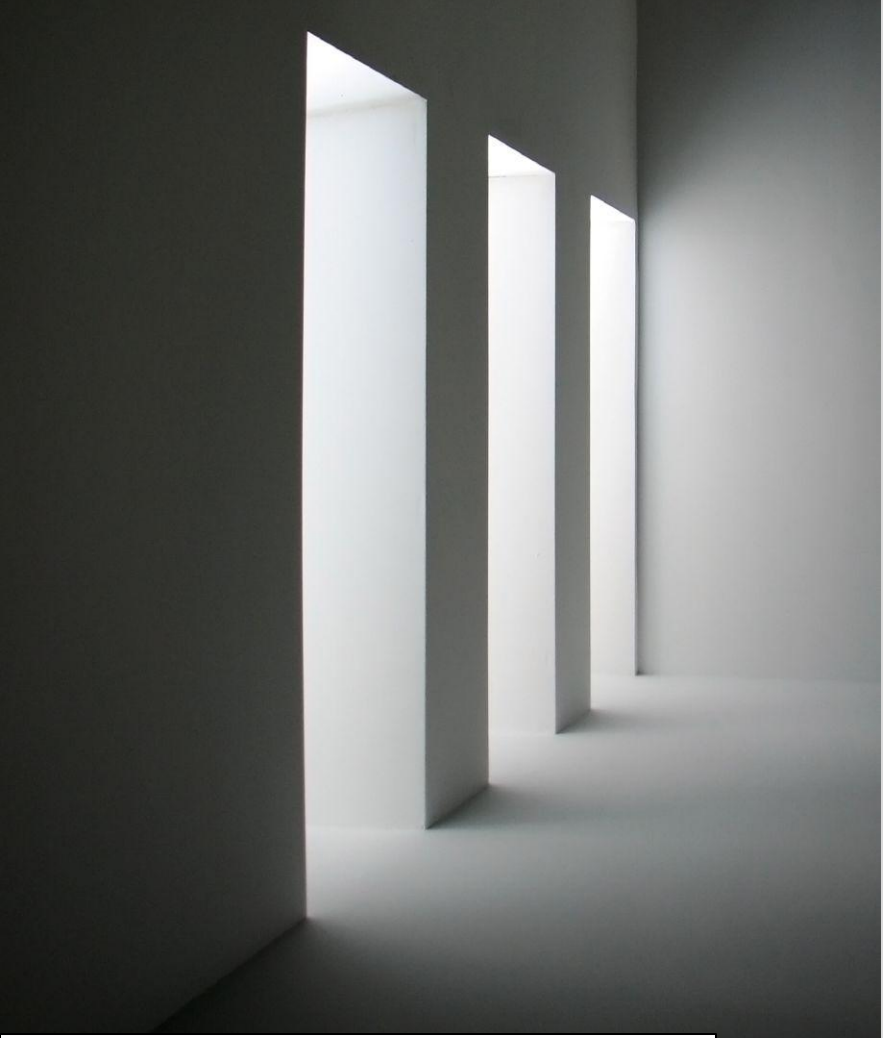
- New lines are your code’s paragraph breaks
- Don’t use them too much or too little

CC-by-2.0 image of bricks by Joost Markerink: https://flic.kr/p/CzNd9S

Be consistent

- Pick styles early
- Update your code
- Keep enforcing them

Why do any of this?



CC image of doorways by MaZzuk: https://flic.kr/p/2jTdEk

Disability can be invisible

@juliaferraioli

Google Open Source

CC image of lightswitch by Paul Comstock: https://flic.kr/p/3g9Y74

Disability isn’t binary

Image of the Earth from SPAAAAACE

Accessibility is for everyone

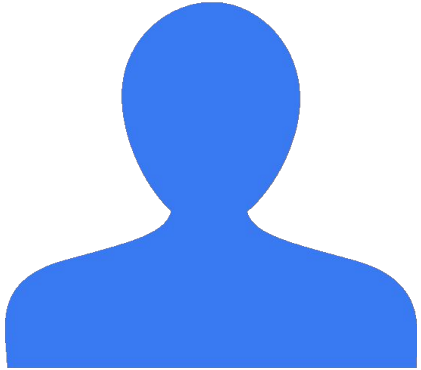
@juliaferraioli

Google Open Source

The curb cut effect

Public domain image of a curb cut from the sidewalk to the street

Curb cut effect



Built for a single population or purpose

Serves the needs of others as well

CC image of a robot with backpack by Morgan: https://flic.kr/p/6UCCUx

Curb cuts for your code

- Improved maintenance
- Better onboarding
- More inclusive environment

Writing accessible Go

- ✓ Organize your code logically
- ✓ Use pronounceable names
- ✓ Be consistent

More info

- Transcript of this talk: http://bit.ly/accessible-go-transcript
- Python Code Style for Blind Programmers: http://bit.ly/2uBWHoY
- Emacspeak: http://bit.ly/emacspeak
- Tools of a Blind Programmer: http://bit.ly/tools-of-a-blind-programmer
- Curb cuts: http://bit.ly/99-curb-cuts

Thank you!

julia ferraioli

jrf@google.com

twitter.com/juliaferraioli