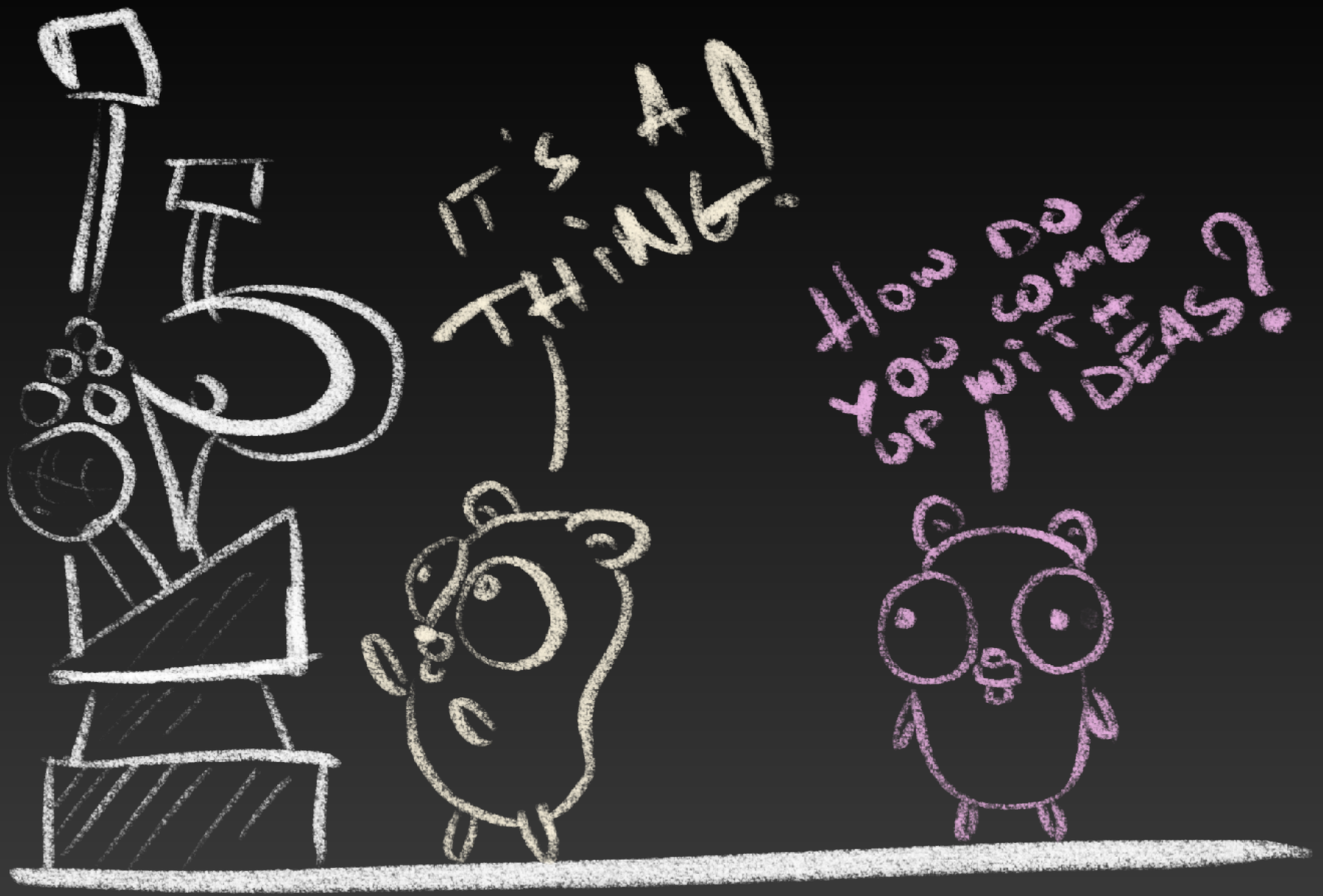
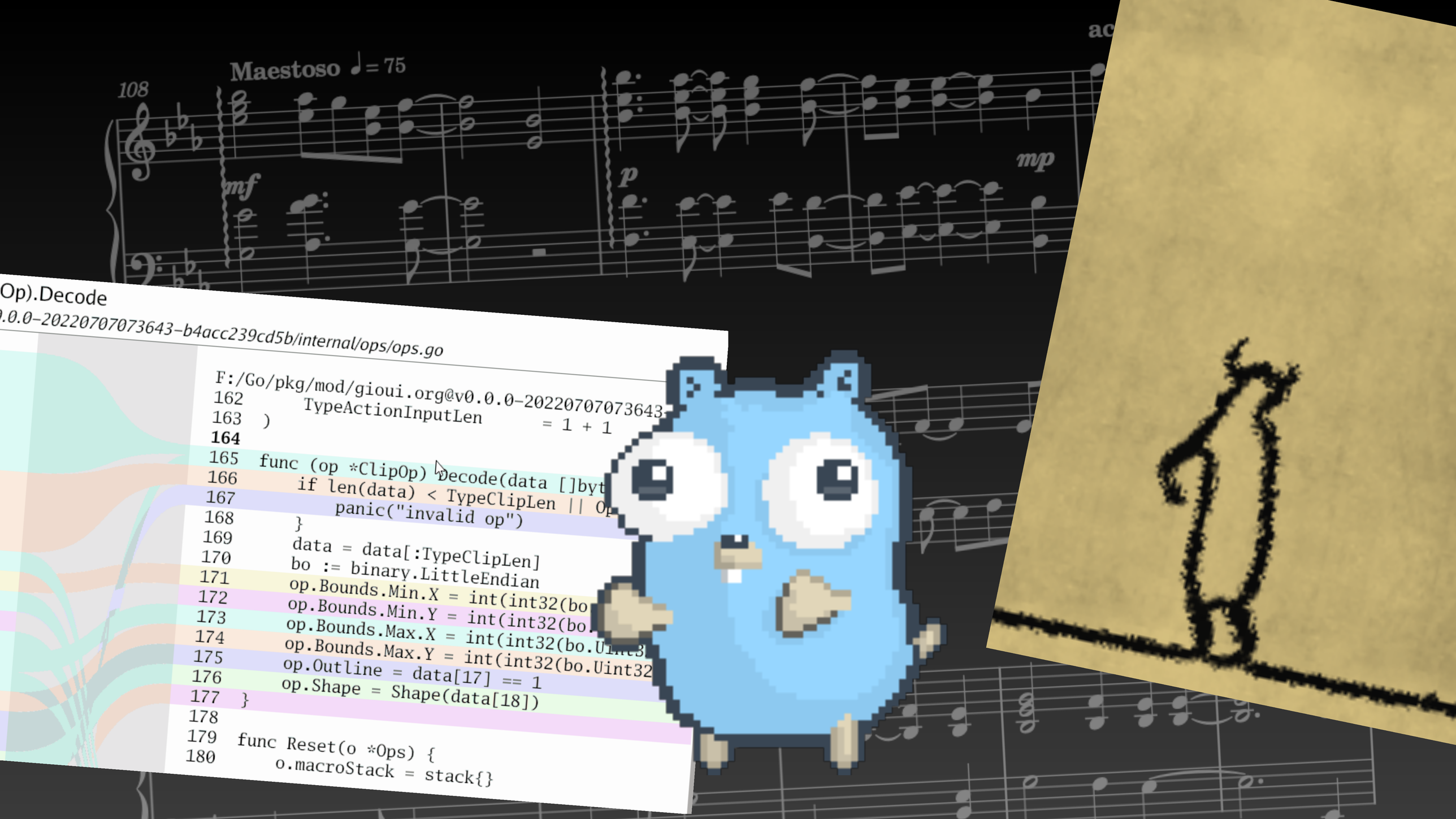
# Where Do Ideas Come From?

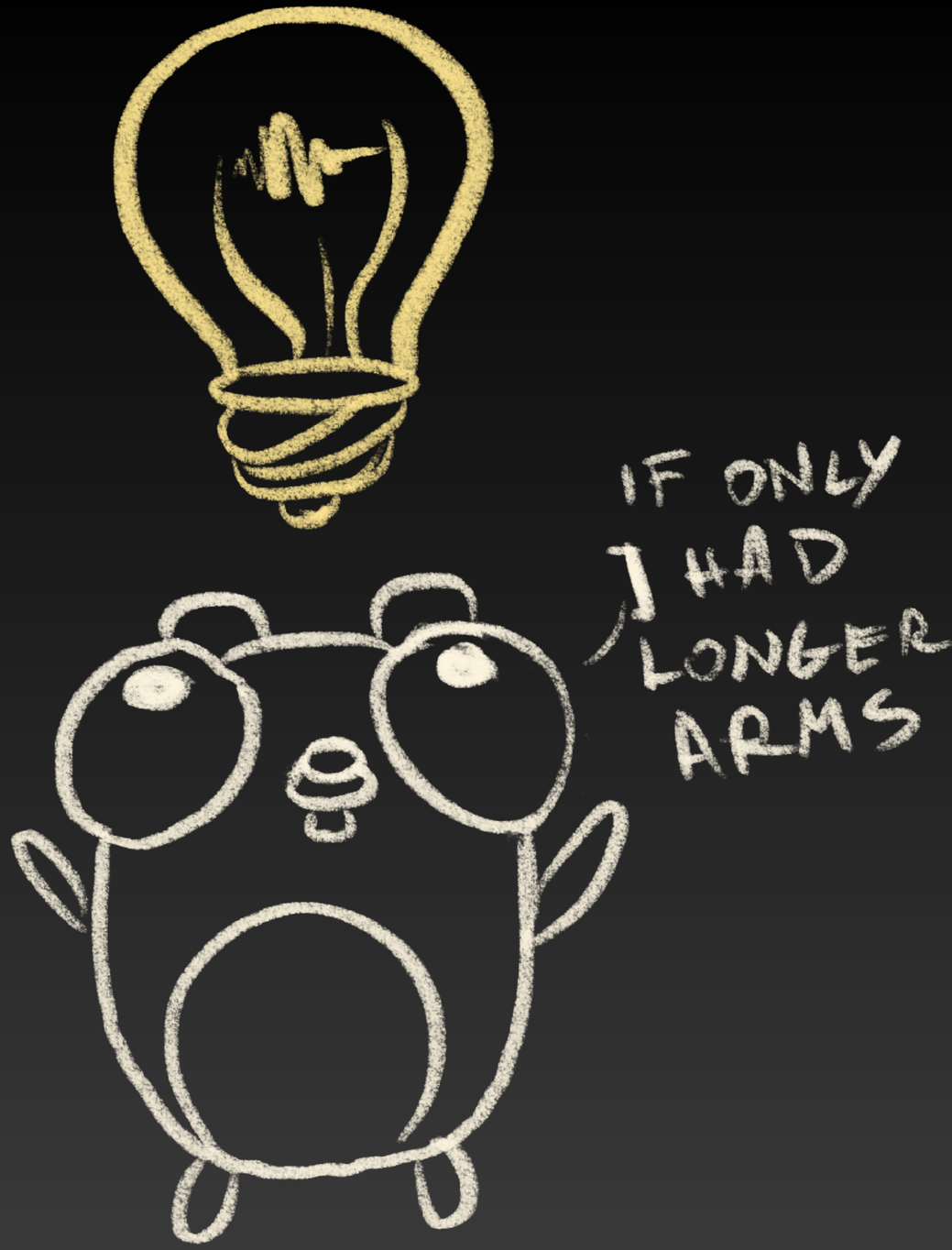


Egon Elbre

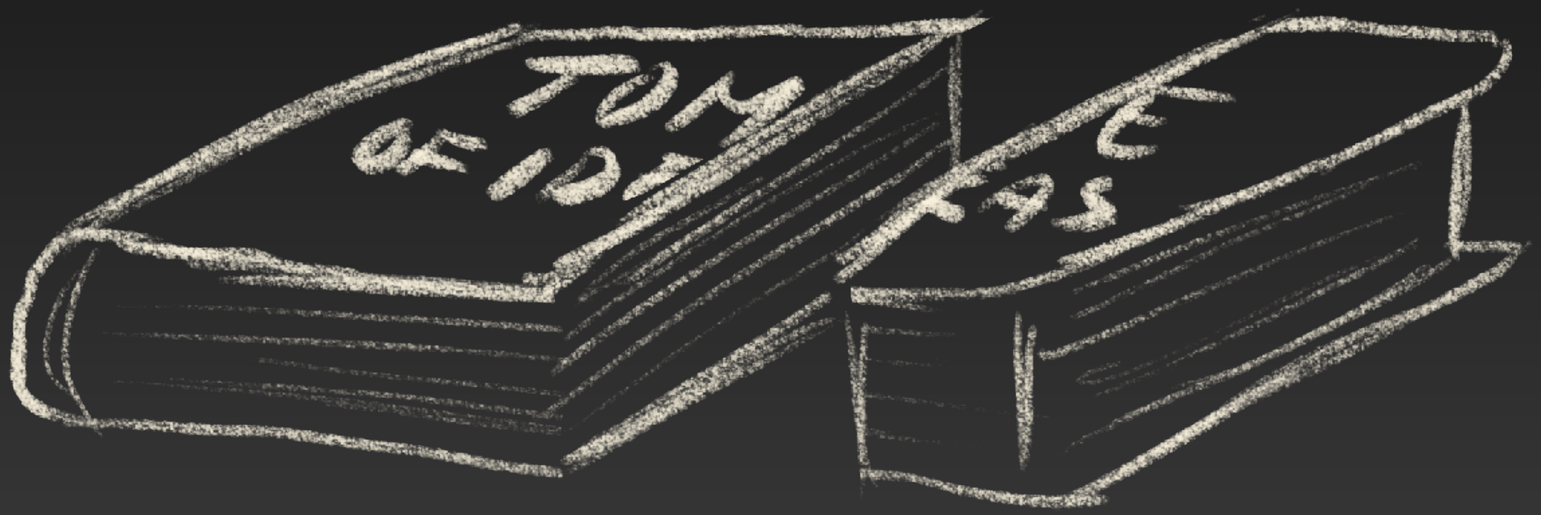

IT'S A THING!

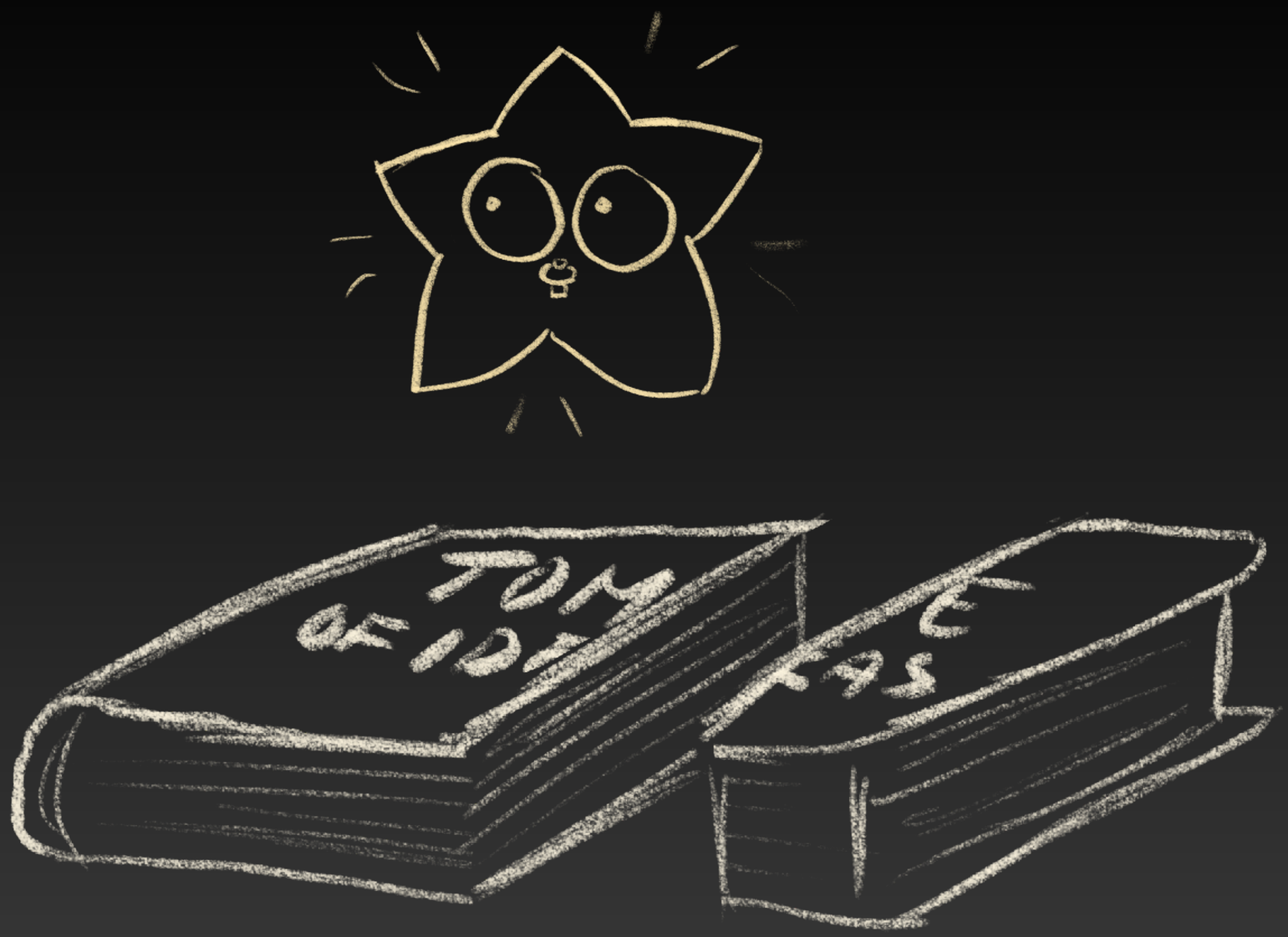

IT'S A THING!
How do you come up with ideas?


Maestoso ♩= 75
108
mf
p
mp
Op).Decode
.0.0-20220707073643-b4acc239cd5b/internal/ops/ops.go
F:/Go/pkg/mod/gioui.org@v0.0.0-20220707073643
162 TypeActionInputLen = 1 + 1
163 )
164
165 func (op *ClipOp) Decode(data []byt
166 if len(data) < TypeClipLen || Op
167 panic("invalid op")
168 }
169 data = data[:TypeClipLen]
170 bo := binary.LittleEndian
171 op.Bounds.Min.X = int(int32(bo
172 op.Bounds.Min.Y = int(int32(bo.
173 op.Bounds.Max.X = int(int32(bo.Uint3
174 op.Bounds.Max.Y = int(int32(bo.Uint32
175 op.Outline = data[17] == 1
176 op.Shape = Shape(data[18])
177 }
178
179 func Reset(o *Ops) {
180 o.macroStack = stack{}


IF ONLY
I HAD
LONGER
ARMS


TOME
OF IDEAS





They are skills and need to be practiced!

Just knowing doesn’t cut it!



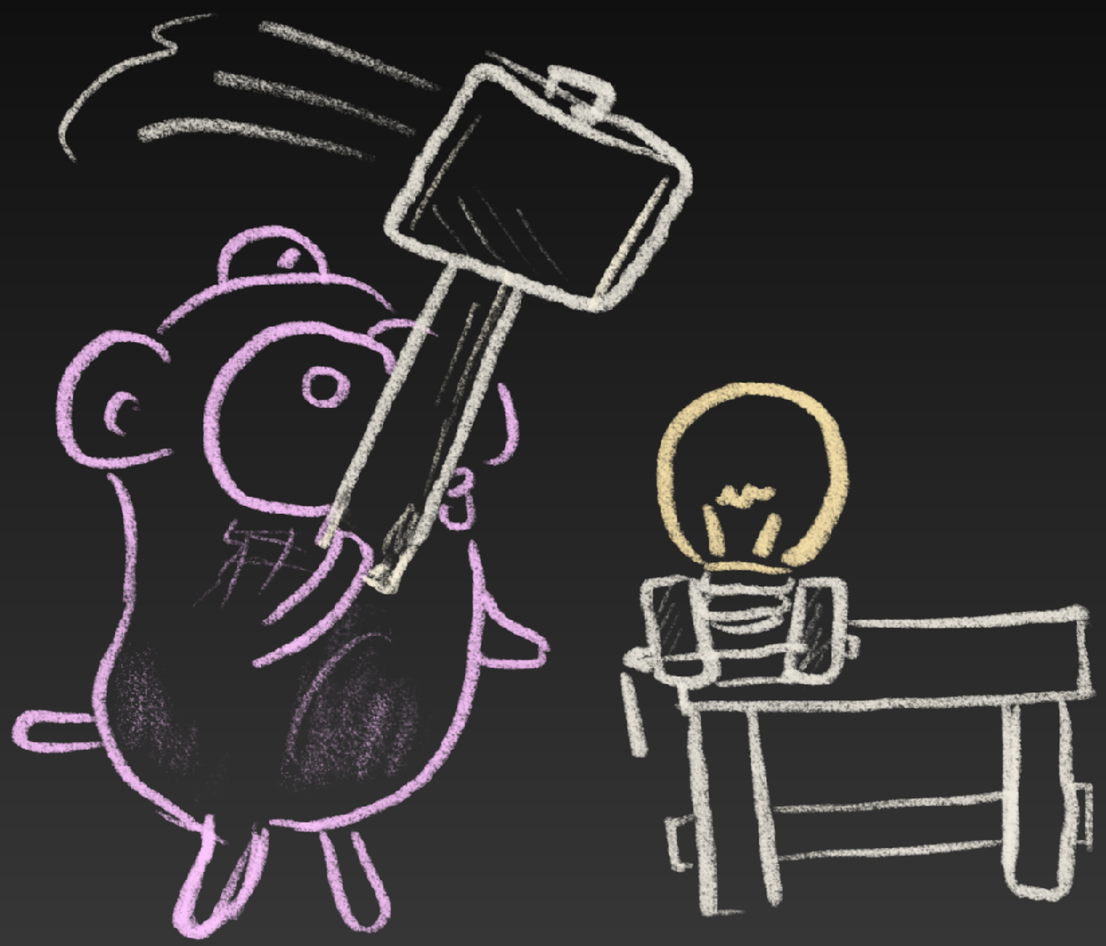
# … but first …



# What's an idea?



# Idea vs. Craft

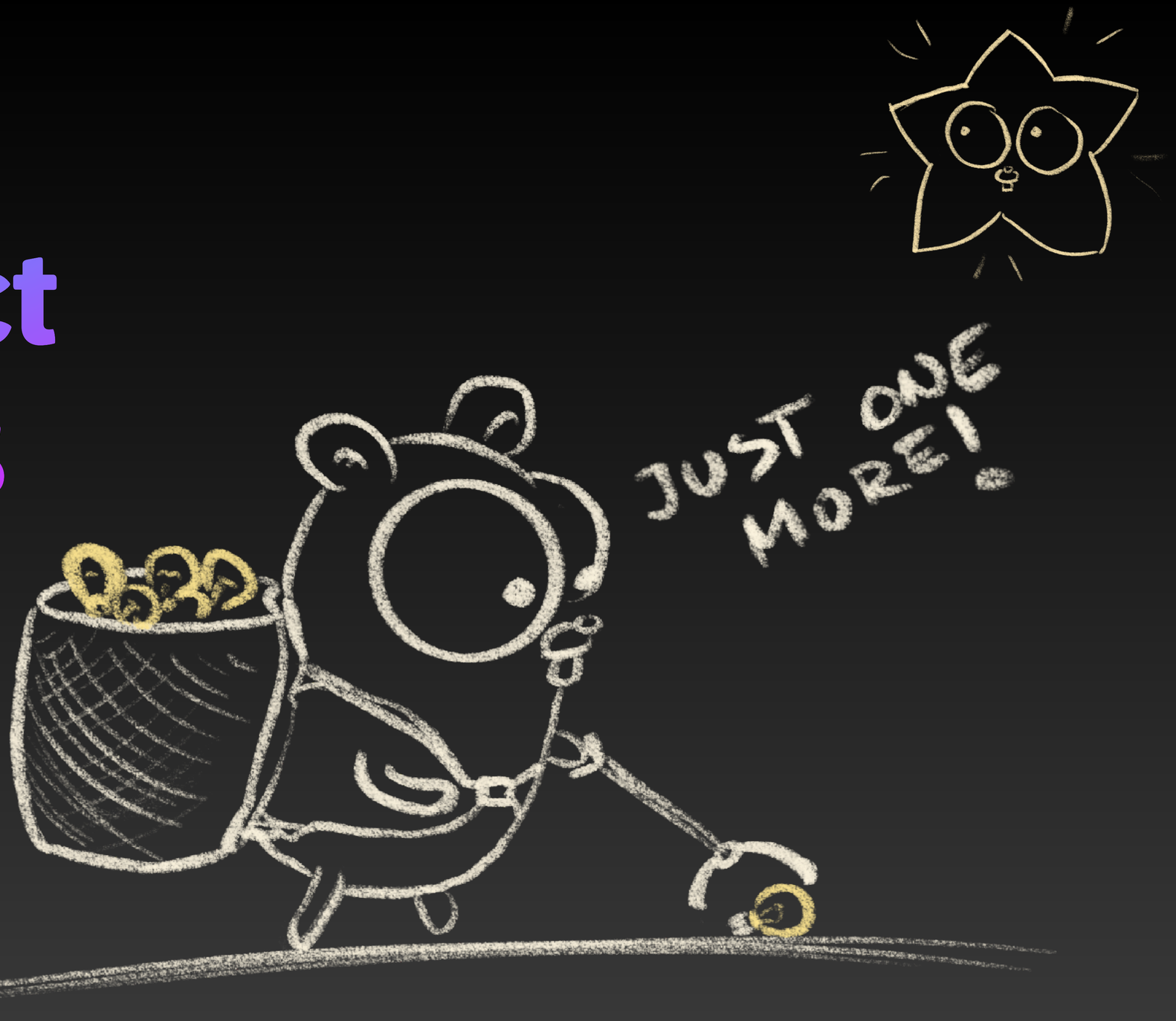


# Part 1: Collecting ideas

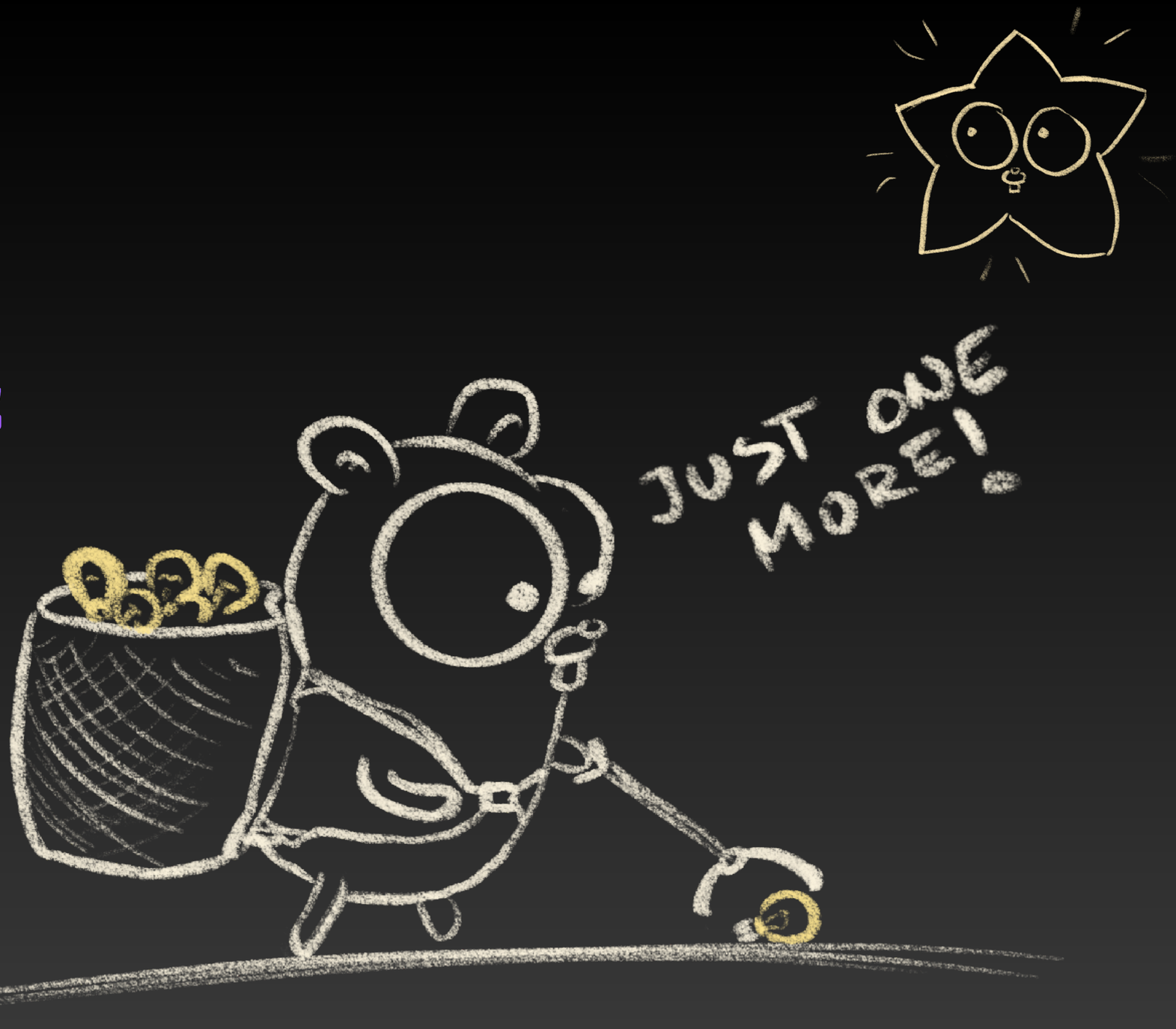
# Ideas come from other ideas...



# ... so, you must collect thousands of ideas...

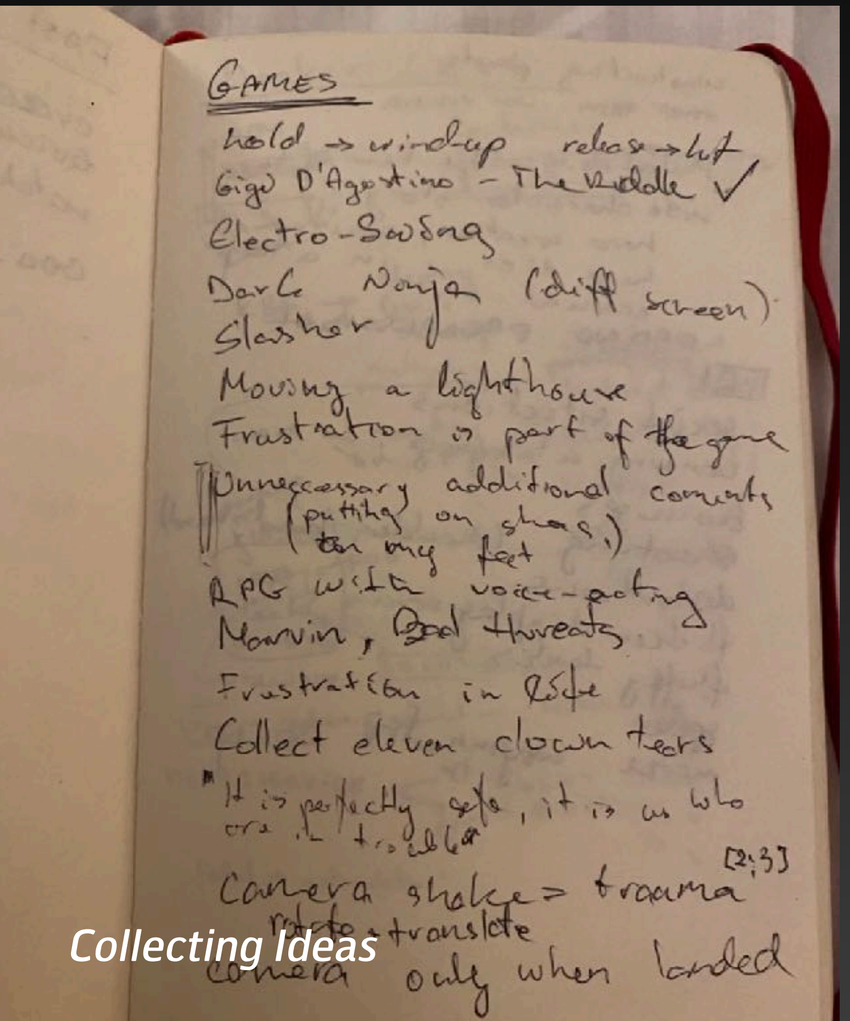


# ... and practice the skill of noticing ideas!



# *my collections*

## Game Ideas

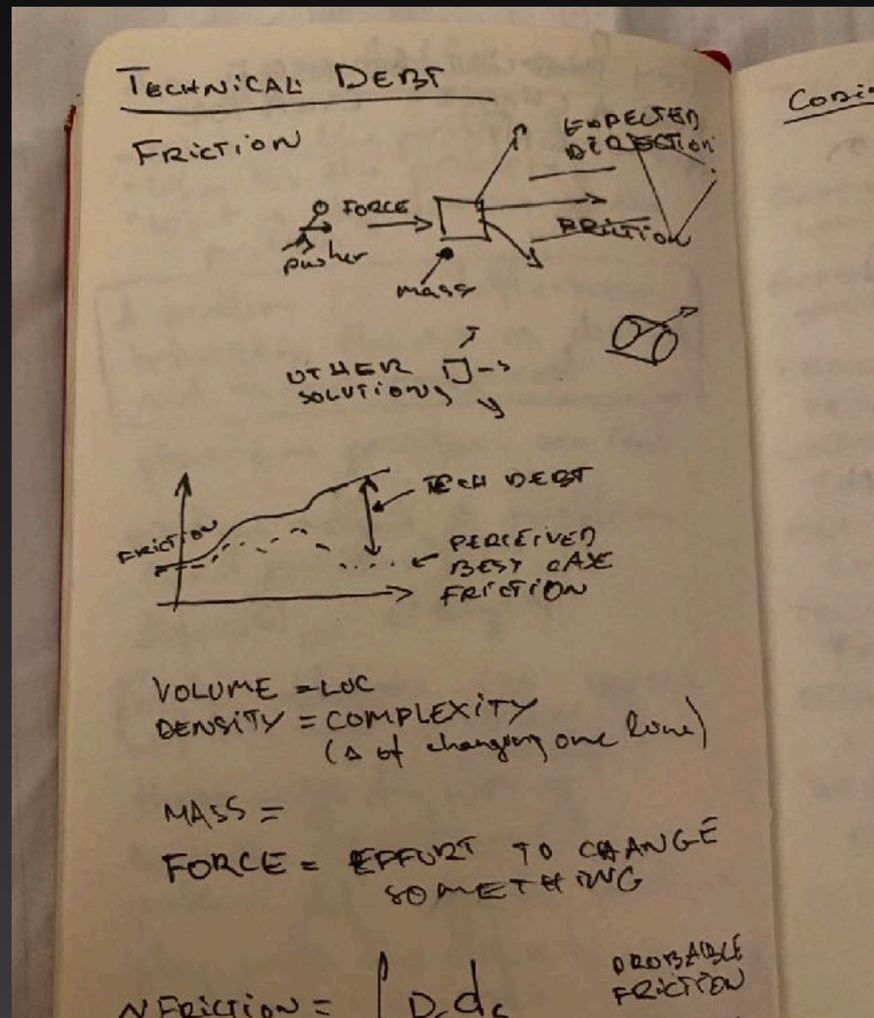



*Collecting Ideas*

## Articles / Talks

- approximating-readability.md
- balancing-coupling.md
- basics-of-usability.md
- beginning-of-programming.md
- benchmarking-ethereum-contracts.md
- checking-architectural-constraints.md
- code-properties.md
- code-structures.md
- data-context-interaction.md
- five-percent-heuristic.md
- hungarian-notation-revisited.md

## An Article



# Collect different things

| **Game Design** | **Algorithmics** | SW. **Architecture** | **UX** |
|---|---|---|---|
| Storylines | Algorithms | Structures | Personas |
| Art Styles | Data Structures | Frameworks | User stories |
| Characters | Paradigms | Principles | Accessibility |
| Places | CPU details | Tools | Cognetics |
| Mechanics | Memory details | Issues | Affordances |
| Technical impl. | Techniques | Forces | Signifiers |
| … | … | … | … |

# example: Frustration

github.com/loov/gorelo

```
gorelo apply gorelo.rules
```

```
server/server.go <-
    Server#handleRequest=HandleRequest
    Server#opts=Opts

# --- db package ---

db/db.go <-
    DB
    dbOptions=Options
    newDB=New

db/db.go <-
    DB#queryRow=QueryRow
    DB#runMigrations=RunMigrations
    DB#opts=Opts
```

# example: Frustration

github.com/loov/ixdiff

```
$ ixdiff tsgo.v0 tsgo.v3
functions: 5044 identical, 232 changed (+17229 relocations), 5 added, 8 removed
total text size delta: +6000 bytes
total spill delta: +181 registers moved
total stack traffic delta: +204 8-byte slots

instruction delta by opcode:
    +362 MOV
    +244 LDR
     +93 BL
    ...

package delta:
       bytes    insts   spills    slots  changed  added  removed  package
       +2784     +699     +102     +118       15      0        2  syscall
        +992     +247      +31      +34        2      2        0  fmt
    ...

top 100 by size delta:
       bytes    insts   spills    slots state     function
       +1216     +304      +48      +56 changed   syscall.forkExec
```

# Collect from different places

Conferences
Other people
Articles
History
Nature
Silence
Random

...




*Collecting Ideas*

# example: Nature

ideas can come from unexpected places



# example: Nature

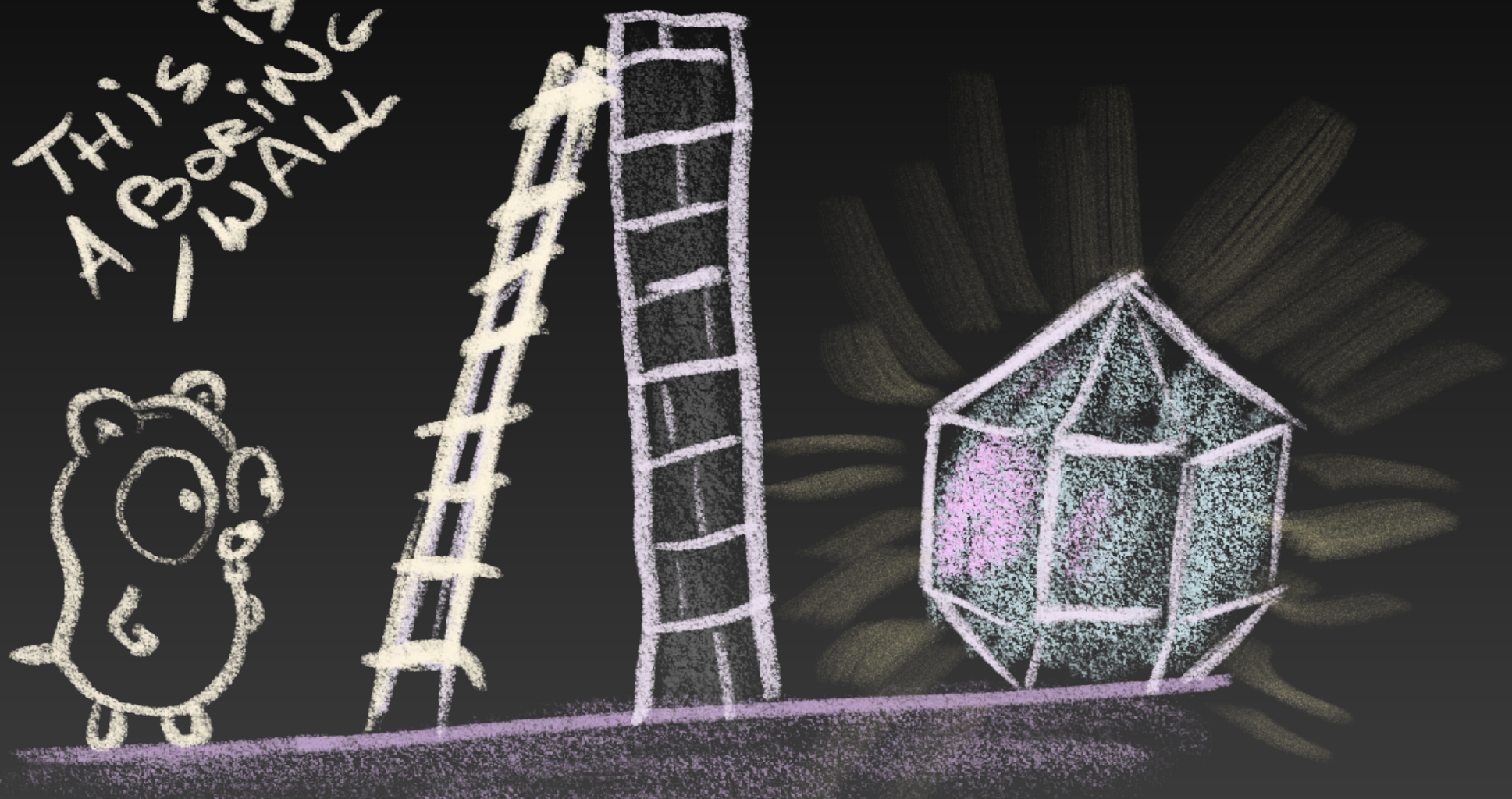
ideas can come from unexpected places



# Tip: more effort leads to easier recall

finding a list of algorithms
taking photos of slides
copy-pasting algorithm descriptions
reading algorithm articles
making notes about algorithms
rewriting algorithms
rewriting algorithms from memory
creating visualisations for algorithms

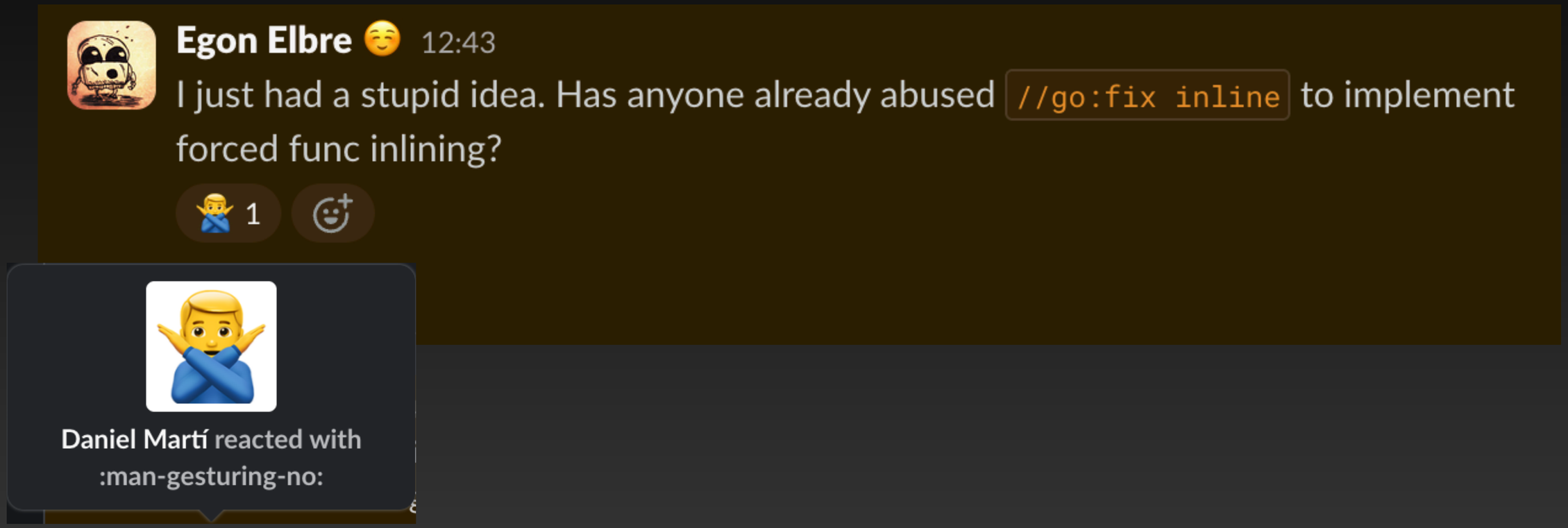
# Pitfall: Filtering early



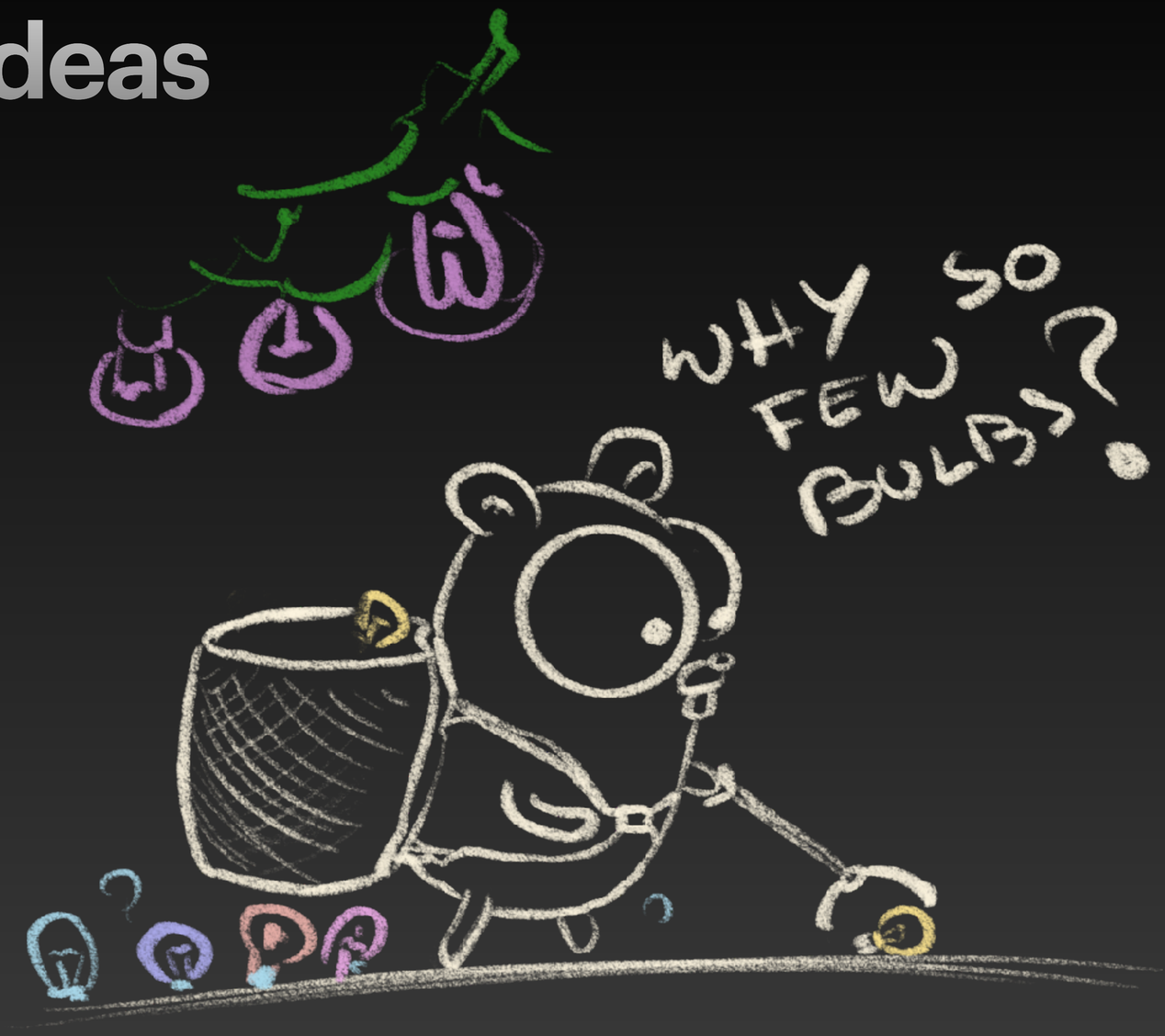

*Collecting Ideas*

# Pitfall: Not collecting bad ideas

#darkarts



# Pitfall: Hyper-focus leads to fewer ideas



# Exercises:

1. Collect 10 ideas, every day, for three months
2. Collect 20+ categories of ideas you could gather
3. Practice collecting different ideas

# Part 2: Analysing ideas

# Ideas come from understanding ideas.




*Analysis and organising*

# Ways to analyse

**Divide into parts** - *divide the idea into smaller sub-ideas.*

**Examine aspects** - *pay attention to only a specific attribute or property of the idea.*

**Change the environment** - *look at the idea by changing the context where it appears.*

**Change the perspective** - *look at the idea from another person's perspective.*

**Find relationships** - *find ideas that fit together.*

**Compare and contrast** - *compare two similar ideas and find ways how they differ.*

**Trace the origins** - *find out the history or imagine how an idea came to be.*

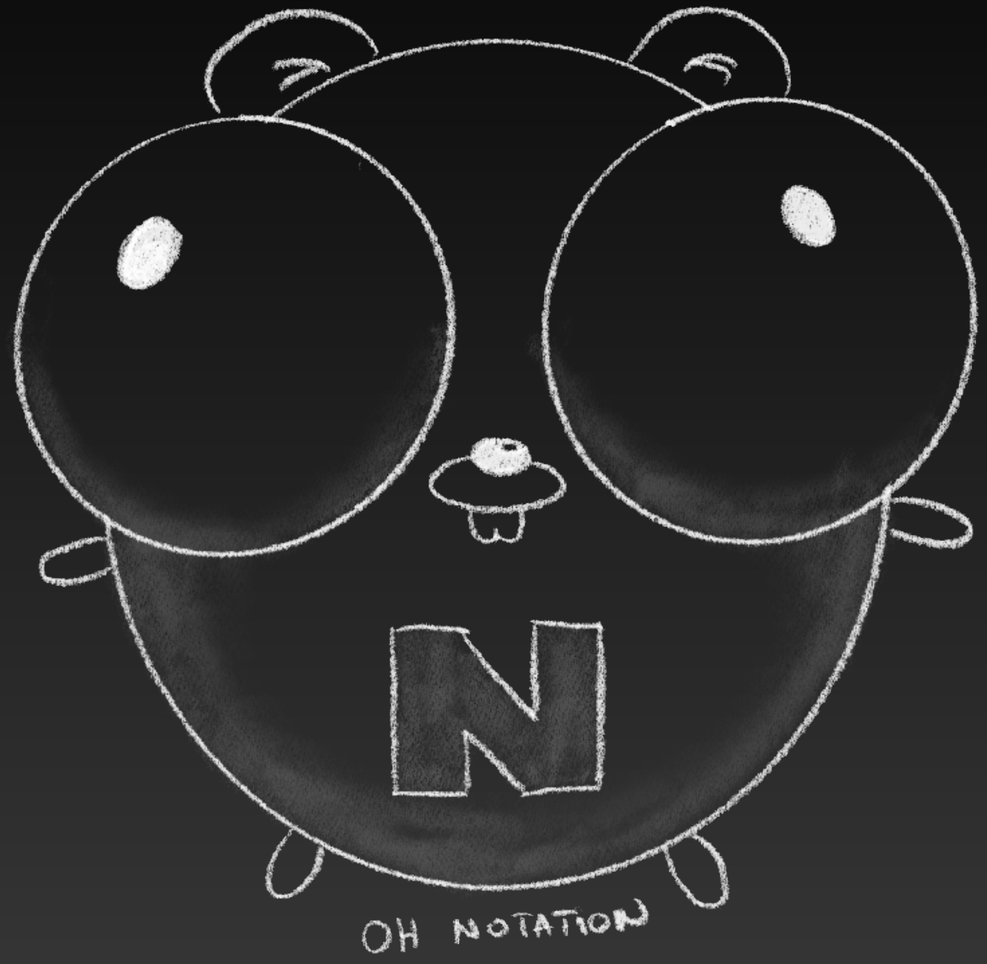
**Predict the future** - *imagine how an idea could change in the future.*

**Recreation and imitation** - *try to reproduce an idea that you admire.*

**Classify and categorise** - *group together or assign labels to similar ideas.*

# Analysis: Examining aspects

| **Code** | **Algorithmics** |
| --- | --- |
| Readability | O-notation |
| Fragility | CPU usage |
| Reliability | Pointer usage |
| Correctness | Memory usage |
| Testability | Memory layout |
| Flexibility | Cache usage |
| … | … |



# tool: “A Deck of Lenses”

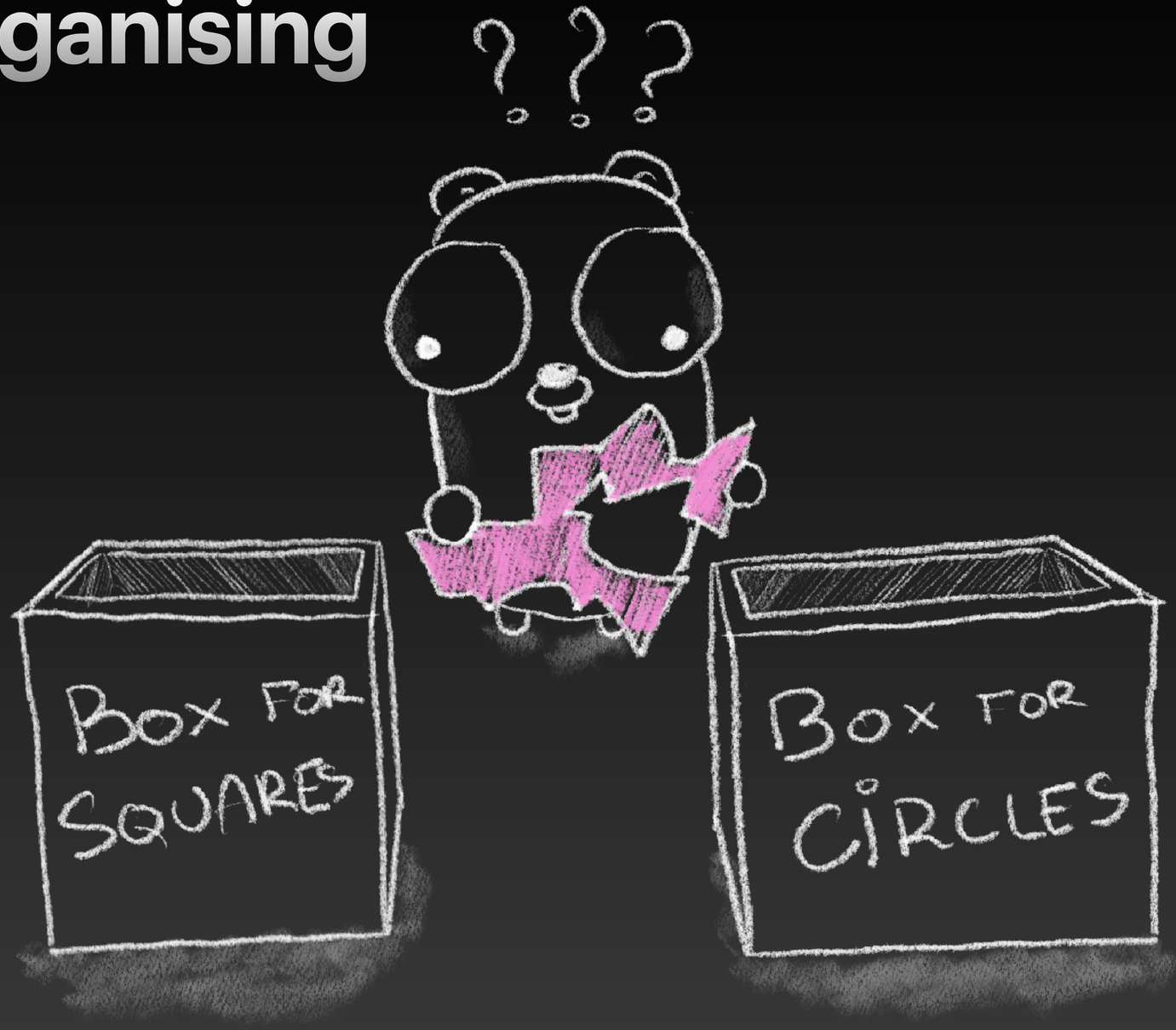






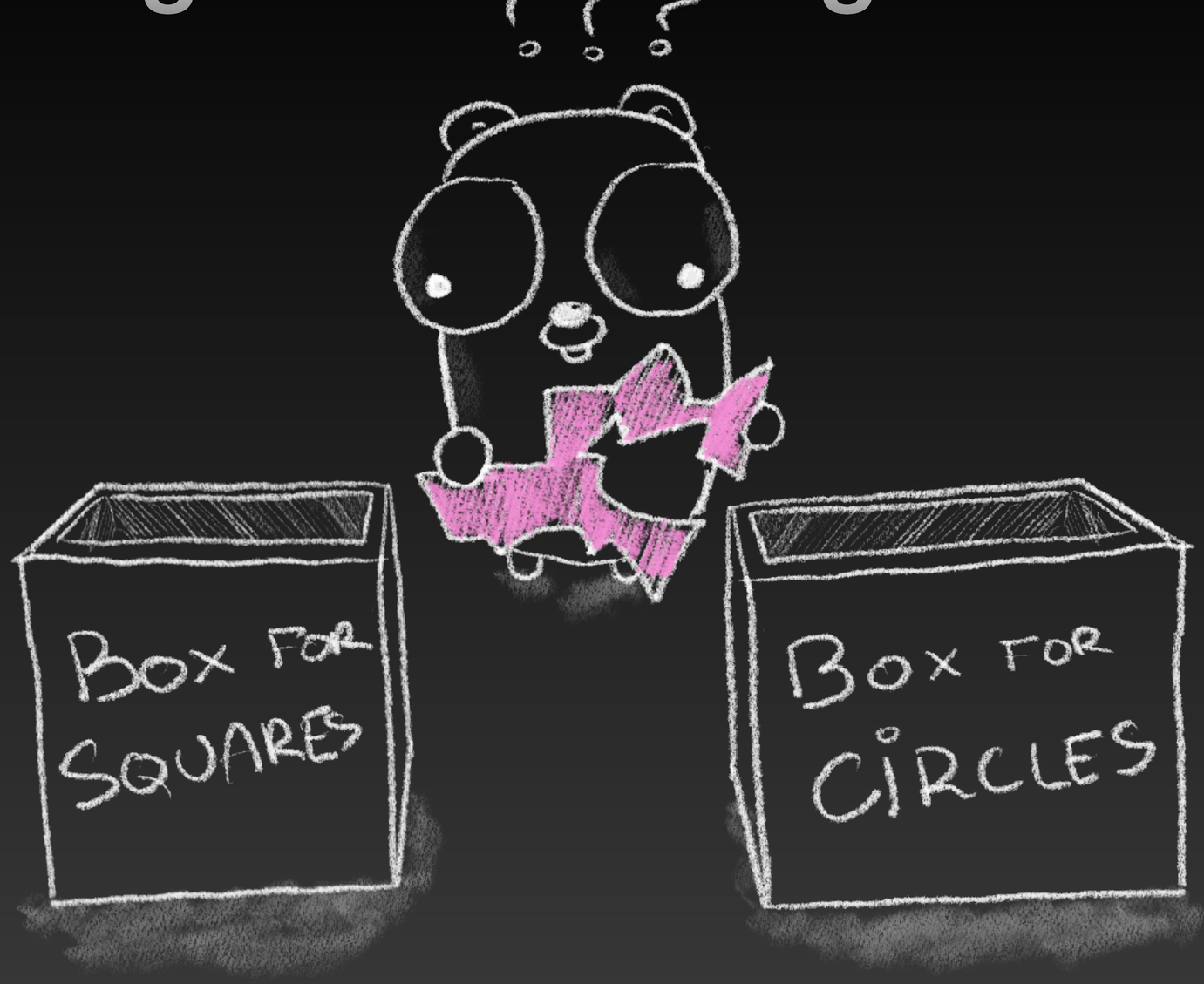
*Analysis and organising*

# Analysis: Organising




*Analysis and organising*

# Pitfall: Organising while collecting




*Analysis and organising*

# Exercises:

1. Collect 20 aspects
2. Analyse 10 ideas using every 20 aspects
3. Practice different analysis methods
4. (Re)Organise your collected ideas once a week

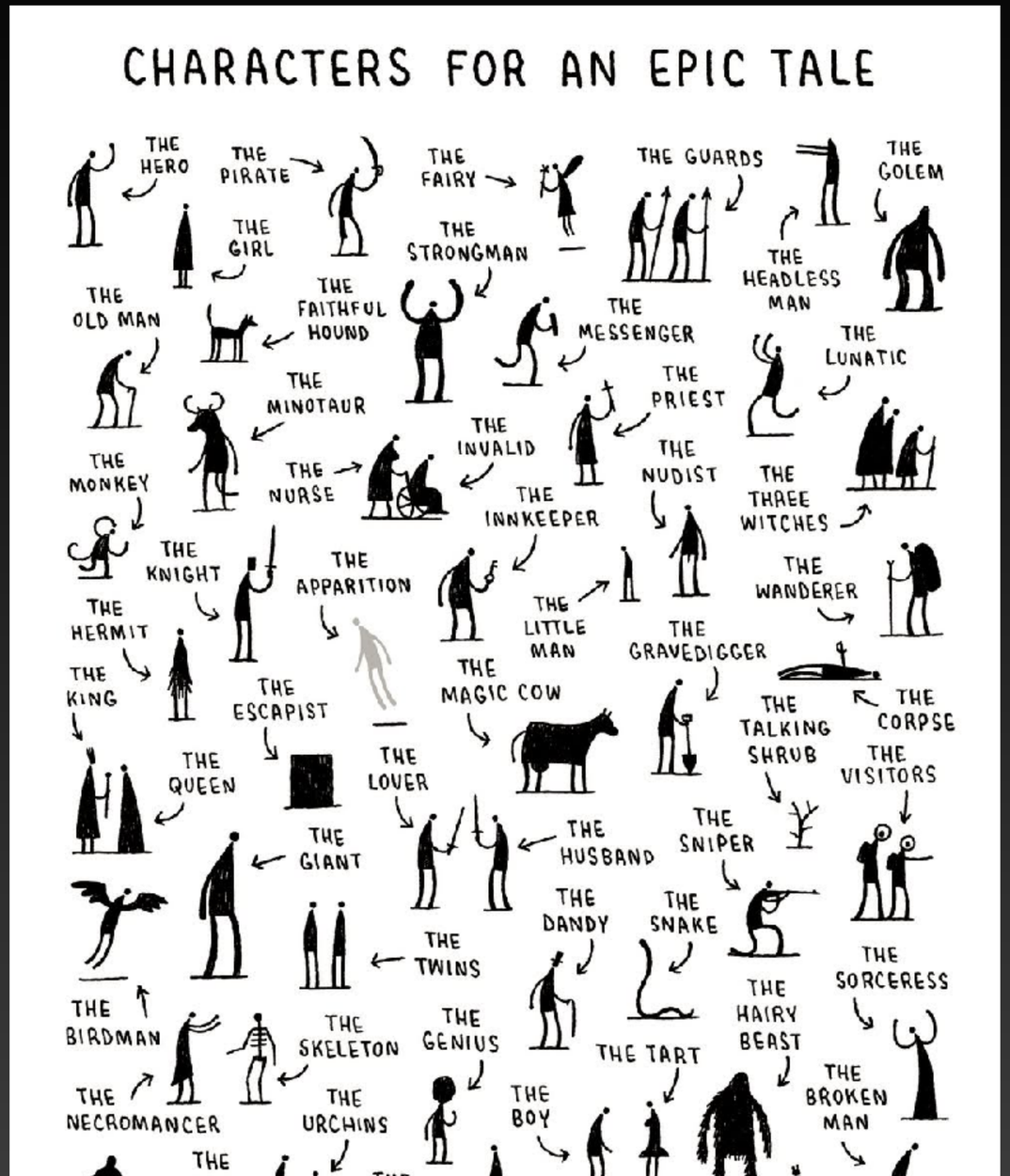
# Part 3: Transforming ideas

# Ideas come from transforming ideas.





# Tom Gauld


CHARACTERS FOR AN EPIC TALE
THE HERO
THE PIRATE
THE FAIRY
THE GUARDS
THE GOLEM
THE GIRL
THE STRONGMAN
THE HEADLESS MAN
THE OLD MAN
THE FAITHFUL HOUND
THE MESSENGER
THE LUNATIC
THE MINOTAUR
THE PRIEST
THE MONKEY
THE INVALID
THE NURSE
THE NUDIST
THE INNKEEPER
THE THREE WITCHES
THE KNIGHT
THE APPARITION
THE WANDERER
THE HERMIT
THE LITTLE MAN
THE GRAVEDIGGER
THE KING
THE ESCAPIST
THE MAGIC COW
THE TALKING SHRUB
THE CORPSE
THE QUEEN
THE LOVER
THE VISITORS
THE GIANT
THE HUSBAND
THE SNIPER
THE DANDY
THE SNAKE
THE SORCERESS
THE TWINS
THE BIRDMAN
THE SKELETON
THE GENIUS
THE HAIRY BEAST
THE TART
THE BROKEN MAN
THE NECROMANCER
THE URCHINS
THE BOY



# SCAMPER

**Substitute** - What if I replace one part of an idea with another part of an idea?

**Combine** - What if I combine two different ideas into a single idea?

**Adapt** - What I borrowed an idea from somewhere else and adjusted it to fit another idea.

**Modify, magnify, minify** - What if I drastically exaggerate aspects of an idea?

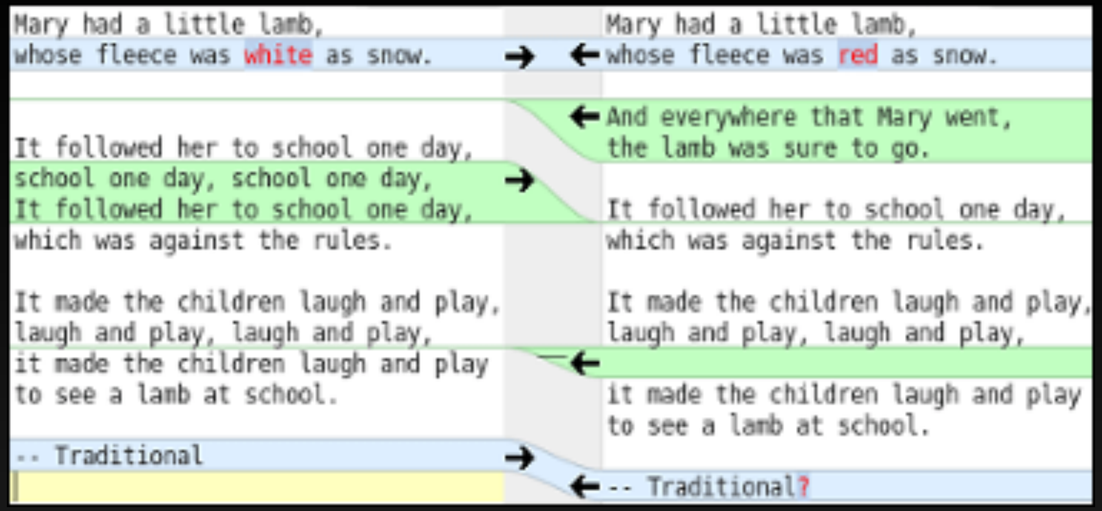
**Put to another use** - What if I took an existing idea and placed it into another context.

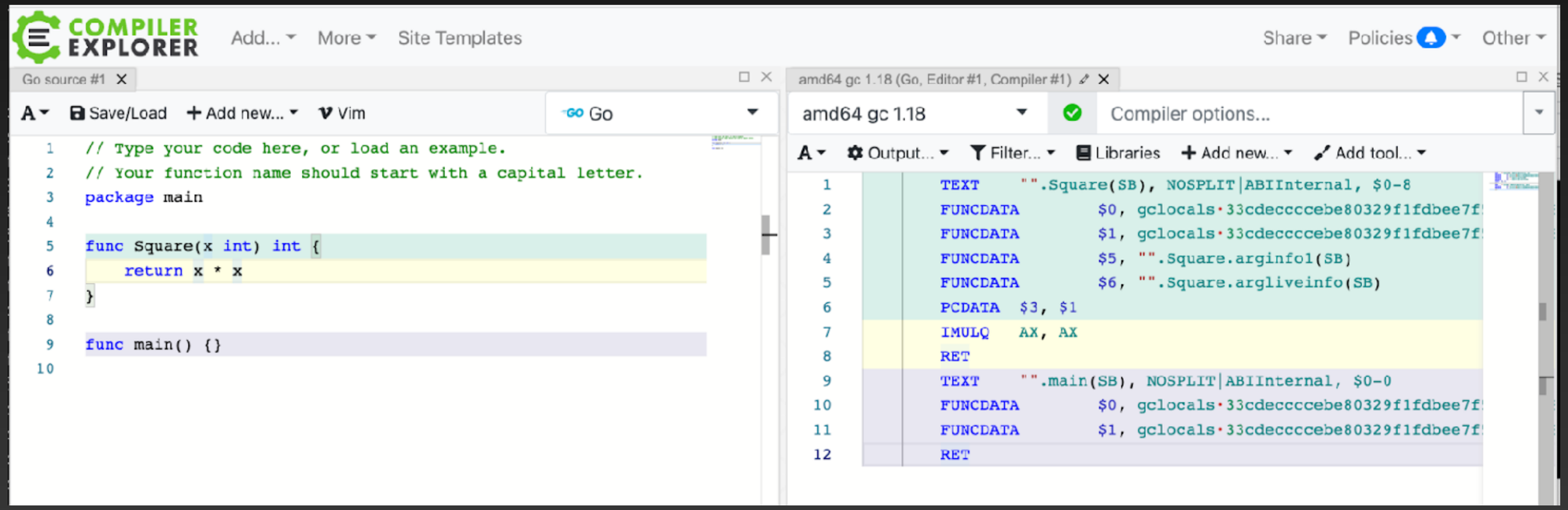
**Eliminate** - What if I removed an important component of the ideas?

**Reverse** - What if I rearrange the whole idea to be in different order or proportions?

*Transforming*

# Combining




*Transforming*

# github.com/loov/lensm

gioui.org/internal/ops.(*ClipOp).Decode

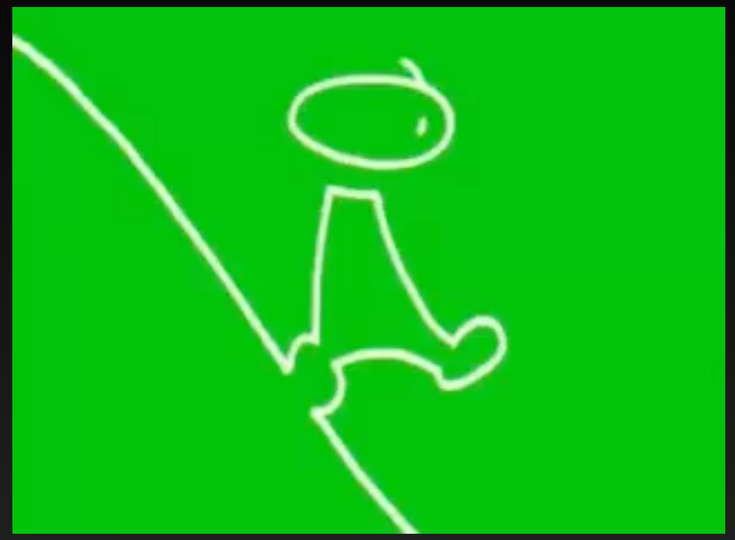
*file: F:/Go/pkg/mod/gioui.org@v0.0.0-20220707073643-b4acc239cd5b/internal/ops/ops.go*

```
CMPQ 0x10(R14), SP
JBE 0x54db70
SUBQ $0x18, SP
MOVQ BP, 0x10(SP)
LEAQ 0x10(SP), BP
MOVQ BX, 0x28(SP)
CMPQ $0x13, CX
JL 0x54db5c
CMPB $0xe1, 0(BX)
JNE 0x54db5c
MOVSXD 0x1(BX), CX
MOVQ CX, 0(AX)
MOVSXD 0x5(BX), CX
MOVQ CX, 0x8(AX)
MOVSXD 0x9(BX), CX
MOVQ CX, 0x10(AX)
MOVSXD 0xd(BX), CX
MOVQ CX, 0x18(AX)
CMPB $0x1, 0x11(BX)
SETE 0x20(AX)
```

```
F:/Go/pkg/mod/gioui.org@v0.0.0-20220707073643-b4ac
162       TypeActionInputLen      = 1 + 1
163   )
164
165   func (op *ClipOp) Decode(data []byte) {
166       if len(data) < TypeClipLen || OpType(data
167           panic("invalid op")
168       }
169       data = data[:TypeClipLen]
170       bo := binary.LittleEndian
171       op.Bounds.Min.X = int(int32(bo.Uint32(data
172       op.Bounds.Min.Y = int(int32(bo.Uint32(data
173       op.Bounds.Max.X = int(int32(bo.Uint32(data
174       op.Bounds.Max.Y = int(int32(bo.Uint32(data
175       op.Outline = data[17] == 1
176       op.Shape = Shape(data[18])
177   }
178
179   func Reset(o *Ops) {
180       o.macroStack = stack{}
```

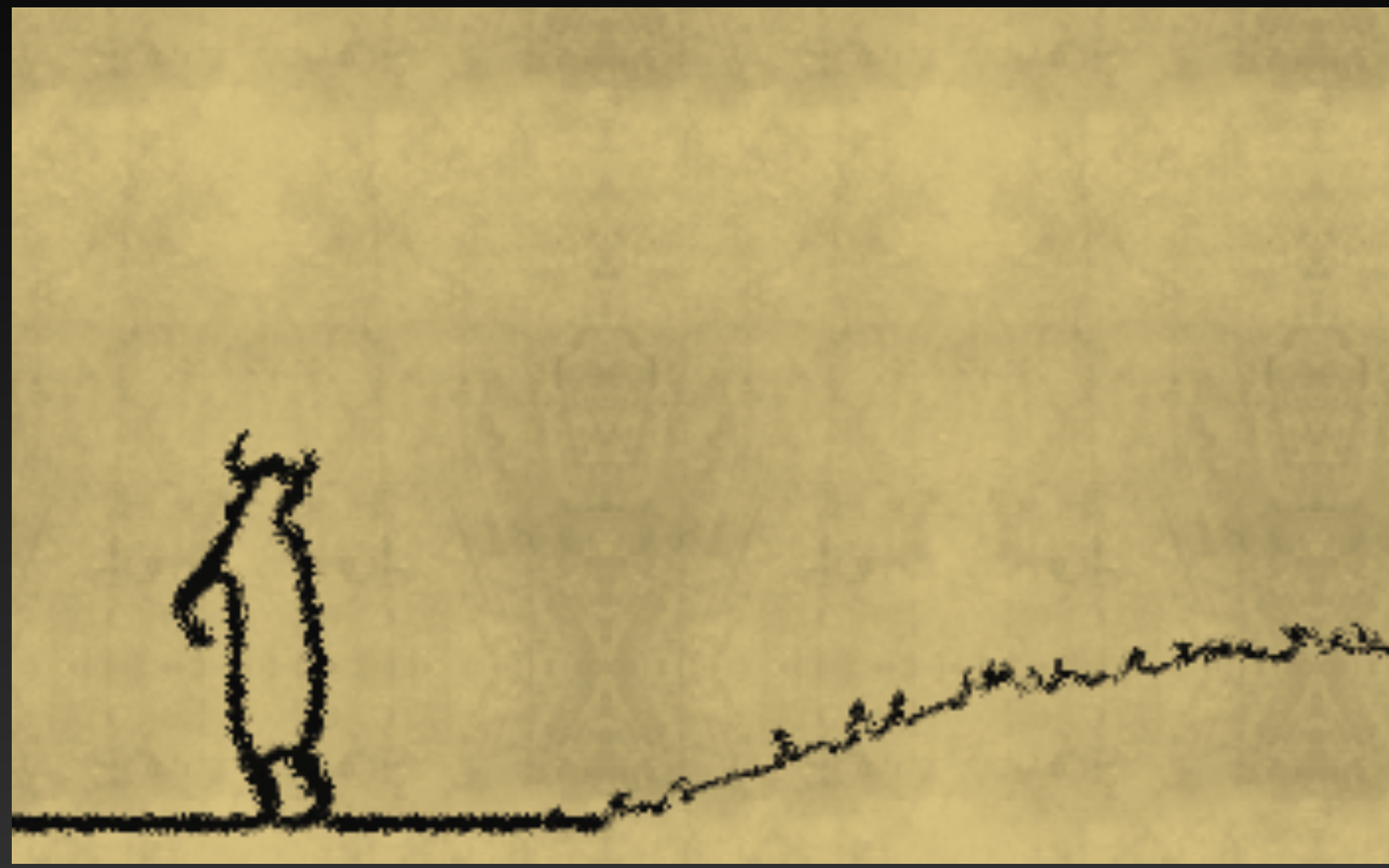
*Transforming*

Gigi D' Agostino: The Riddle






Dancing Plague [from a game jam]

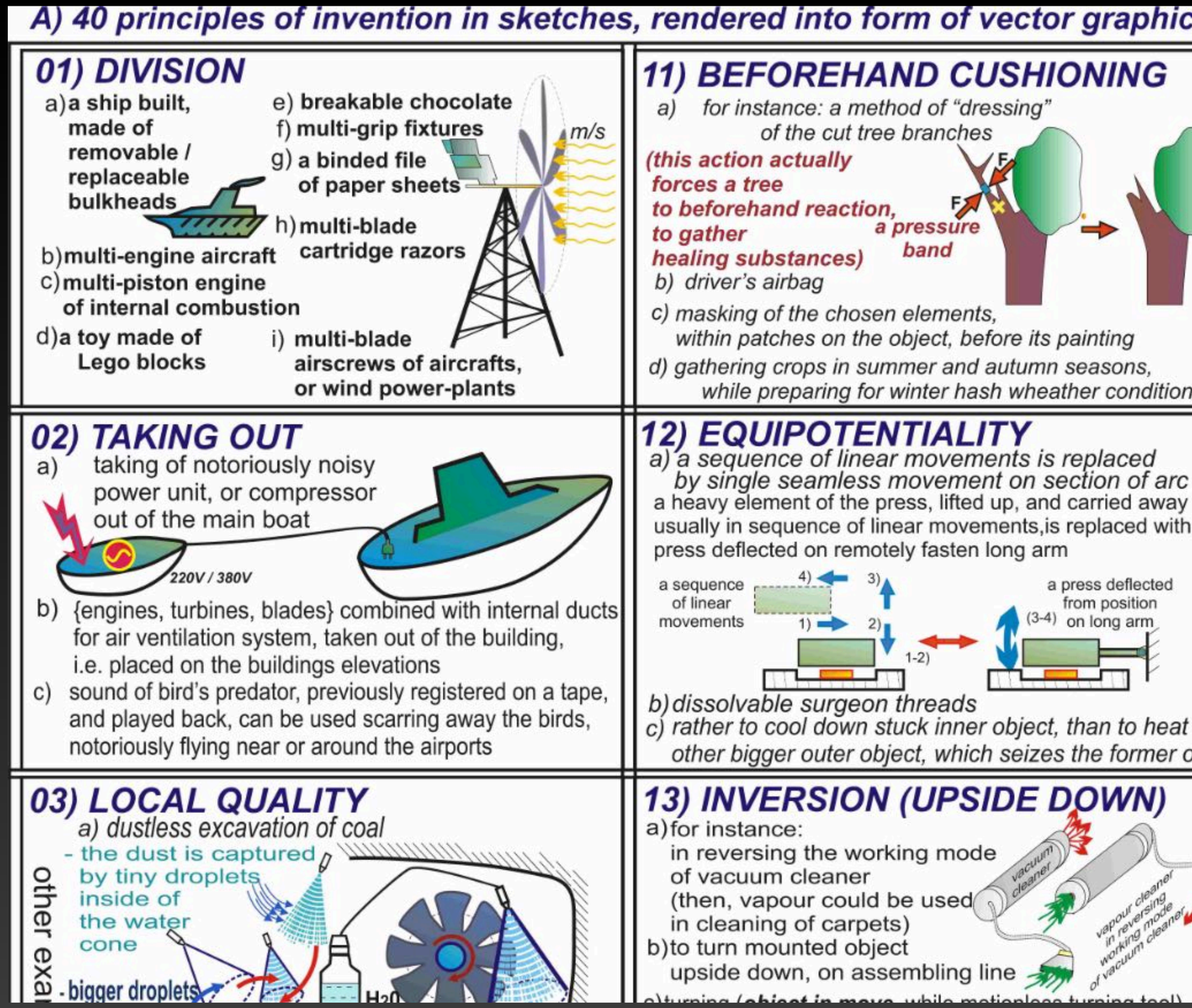


*Transforming*

# TRIZ

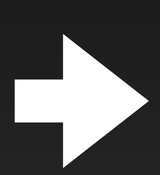
Theory of inventive problem solving

*Transforming*



# example: SIMD-ifying

```
func NTT(f RingElement) NTTElement {
    m := byte(0)
    for len := 128; len >= 1; len /= 2 {
        for start := 0; start < 256; start += 2 * len
            m++
            zeta := zetas[m]
            // Bounds check elimination hint.
            xs, ys := f[start:start+len], f[start+len:
            for j := 0; j < len; j++ {
                t := FieldMontgomeryMul(zeta, ys[j])
                ys[j] = FieldSub(xs[j], t)
                xs[j] = FieldAdd(xs[j], t)
```



```
func NTT8(f RingElement) NTTElement {
    m := byte(0)

    for len := 128; len >= 8; len /= 2 {
        for start := 0; start < 256; start +=
            m++
            zeta := BroadcastUint32x8(zetas[m]

            xs, ys := f[start:start+len], f[sta
            for j := 0; j < len; j += 8 {
                y := LoadUint32x8Slice(ys[j:])
                t := FieldMontgomeryMul8(zeta,
```

*Transforming*

# Patterns and Pattern Languages

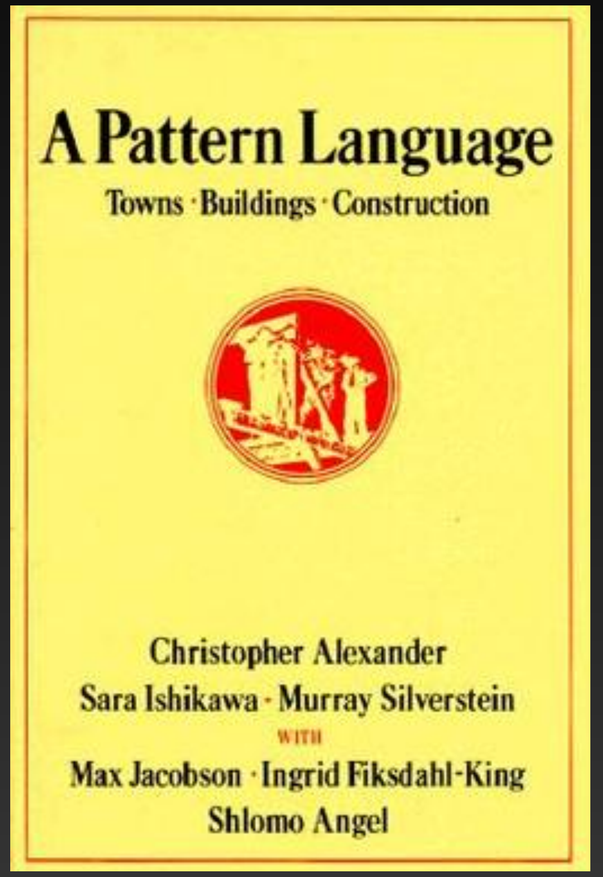
Picture

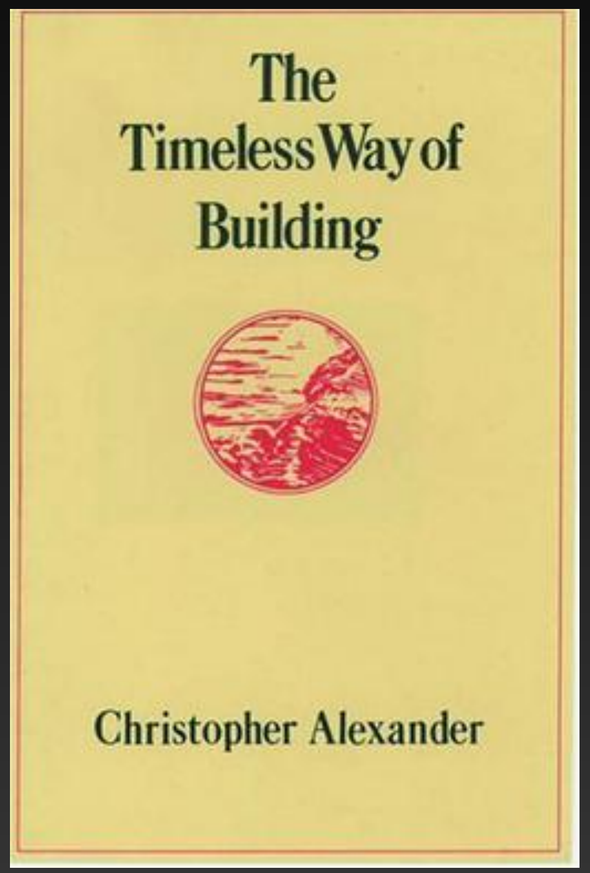
Context

Problem Statement

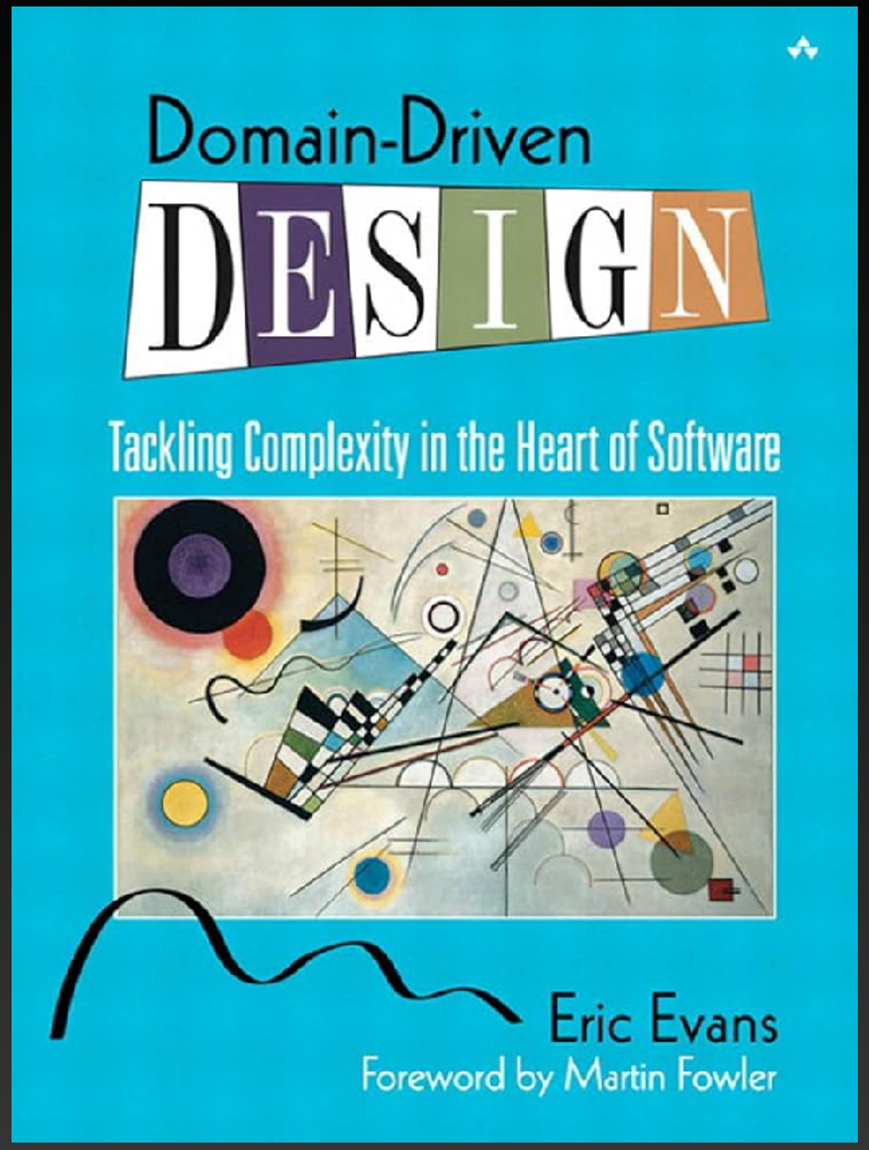
Solution

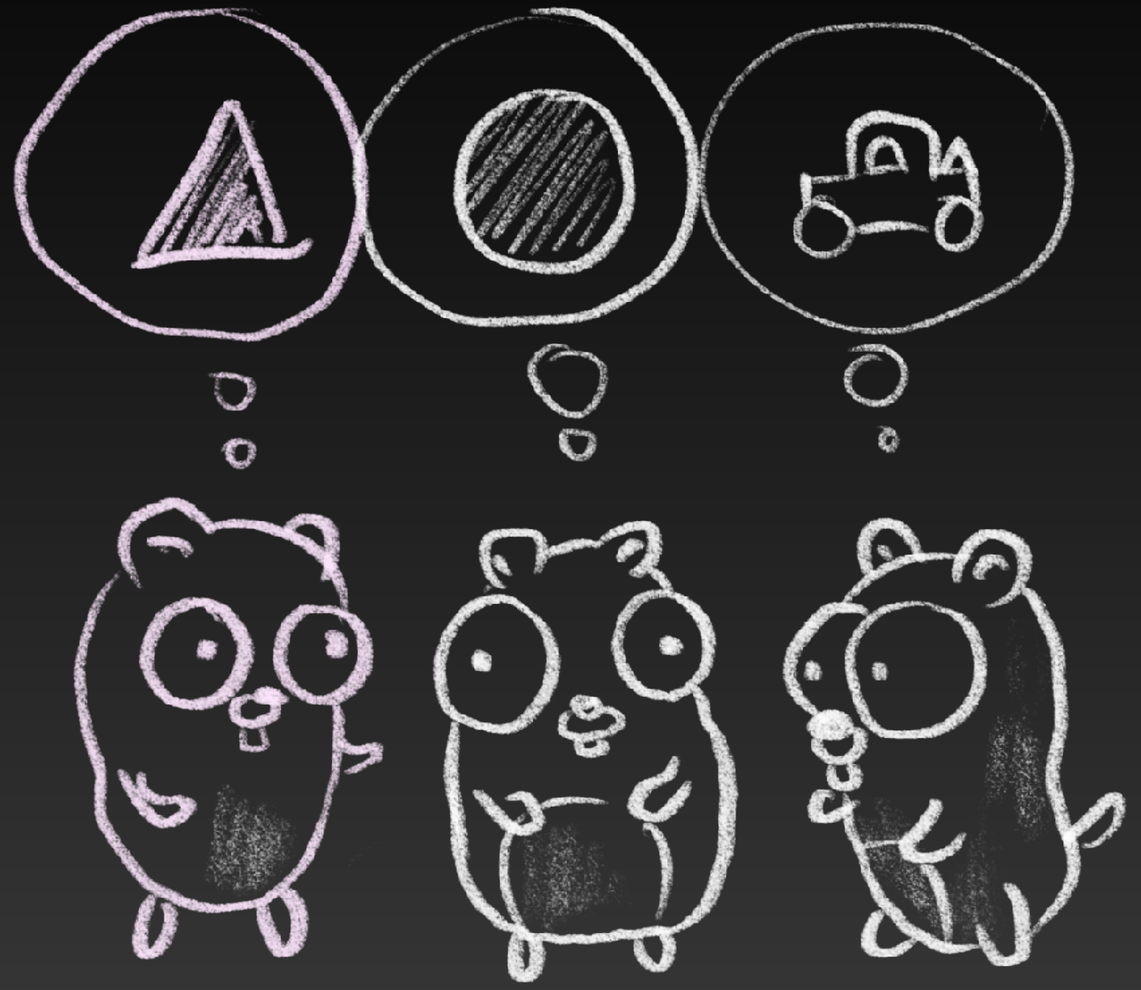
Network Links



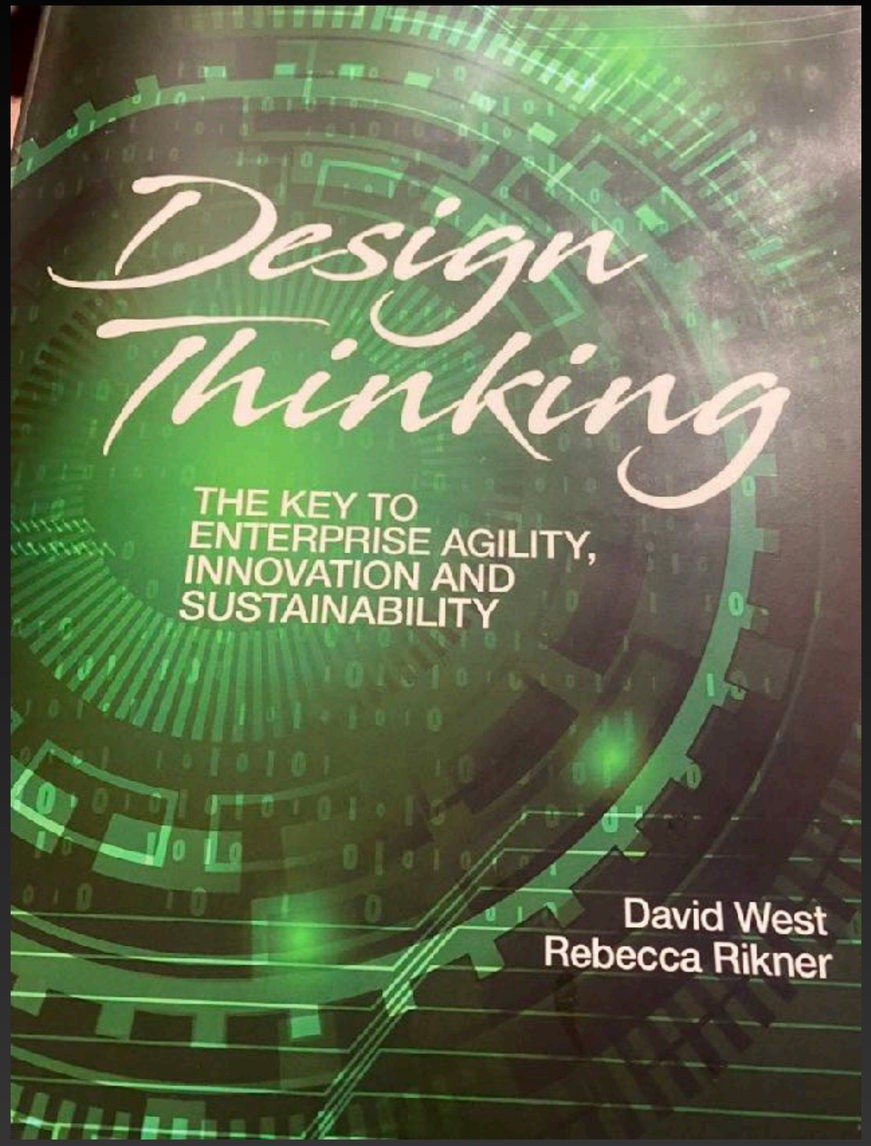



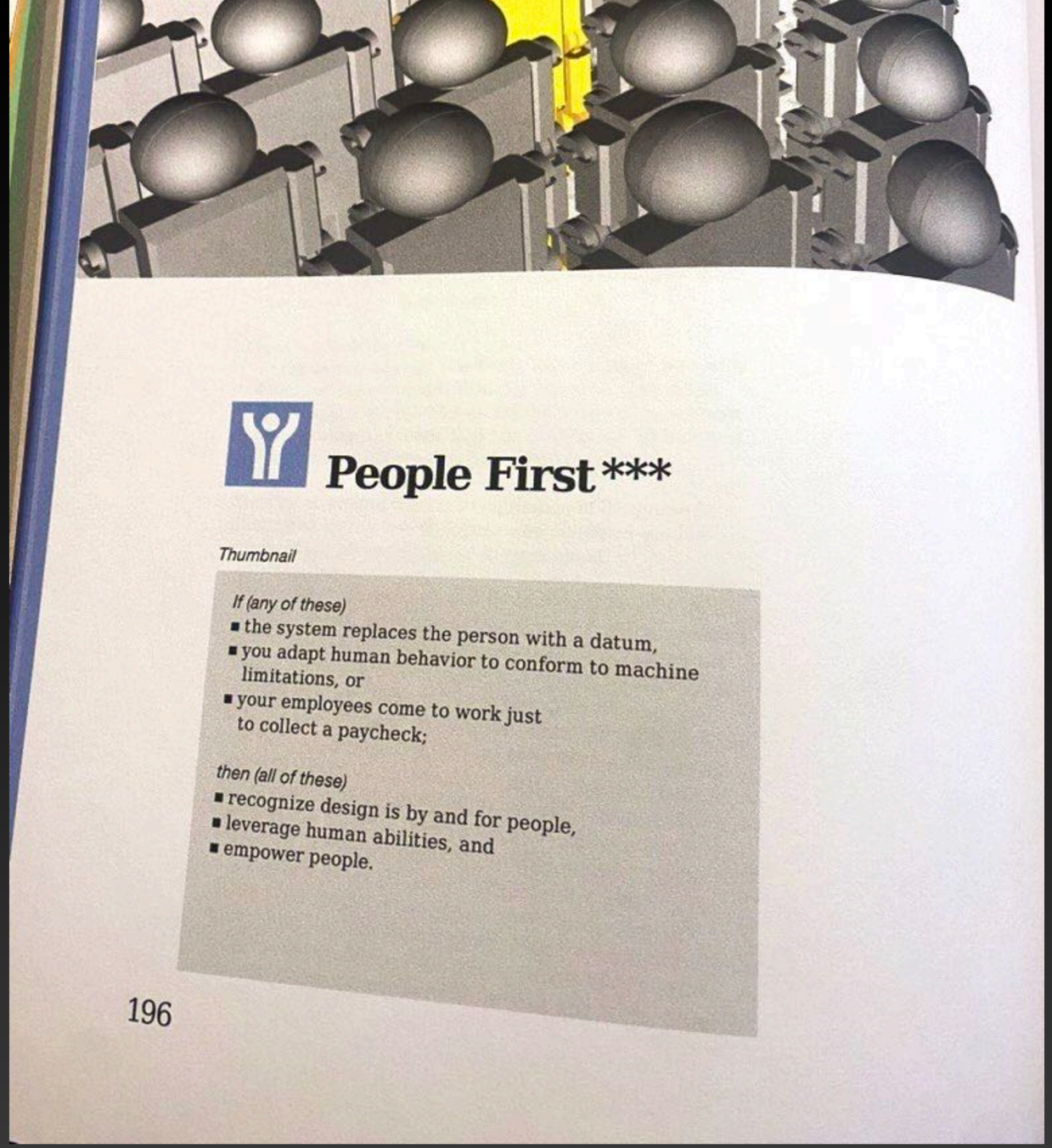
# “Ubiquitous Language”



*Transforming*



## People First***

*Thumbnail*

*If (any of these)*
- the system replaces the person with a datum,
- you adapt human behavior to conform to machine limitations, or
- your employees come to work just to collect a paycheck;

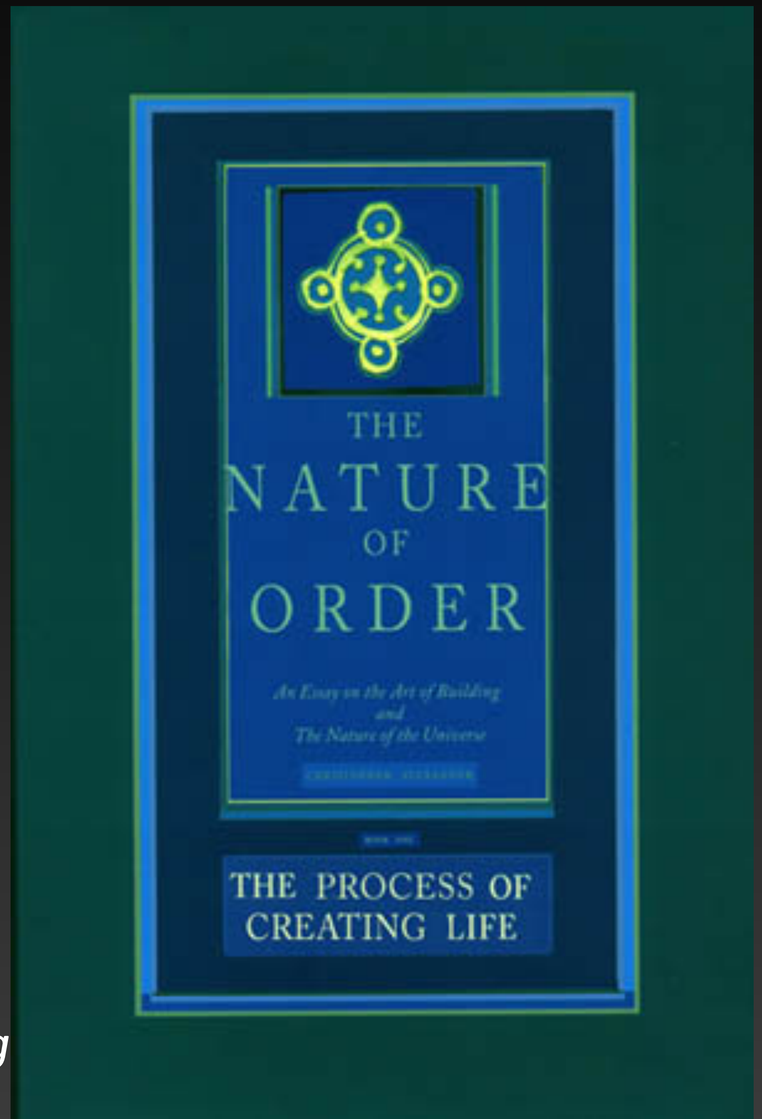
*then (all of these)*
- recognize design is by and for people,
- leverage human abilities, and
- empower people.

196

*Transforming*

# Fifteen Properties in Nature

## *Christopher Alexander*

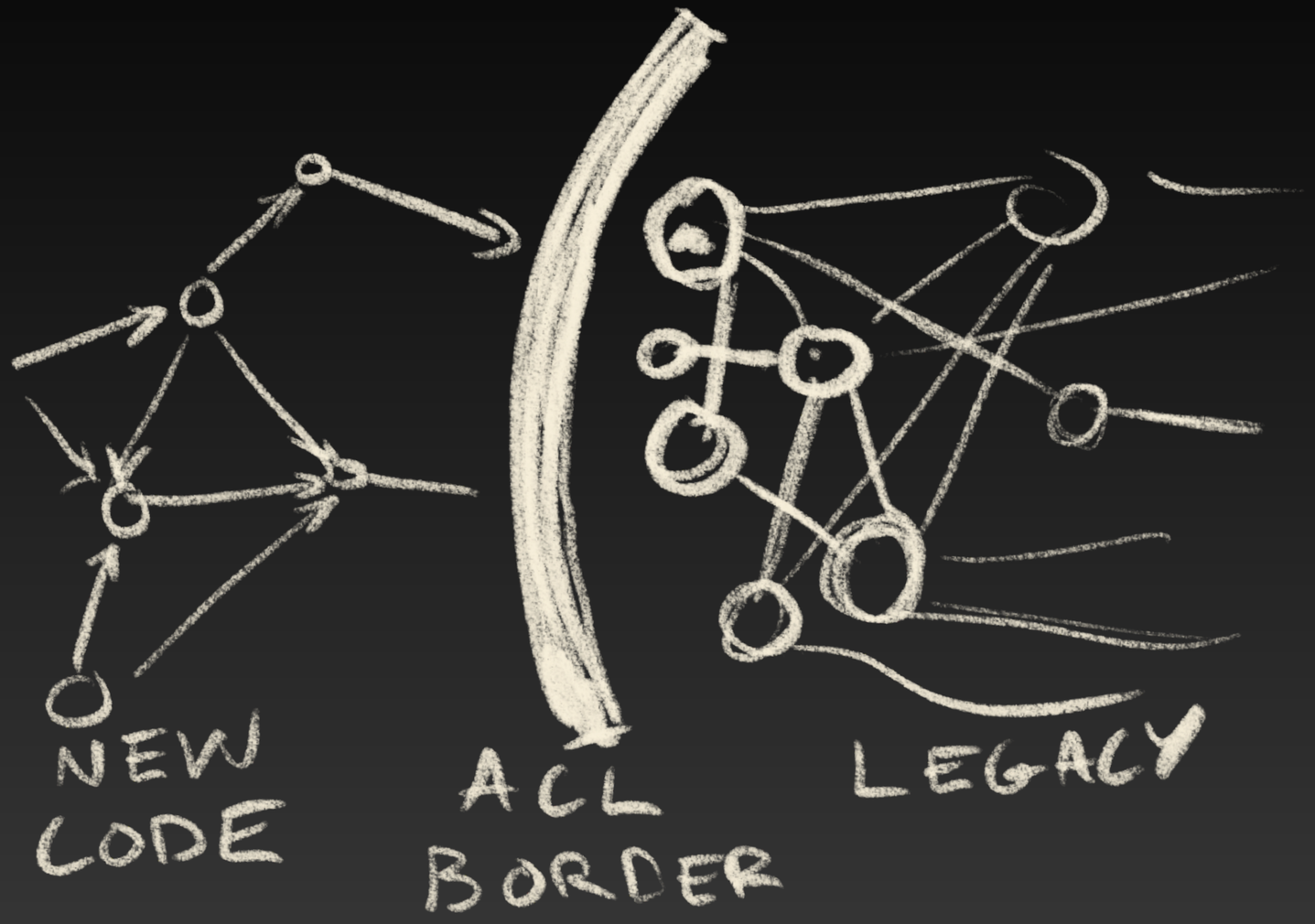






*Transforming*

# *Boundaries:* ***Anti-Corruption Layer*** *from DDD*



# Ambiguous transformations

Oblique Strategies
by Brian Eno

Discover the recipes you are using and abandon them

Go

Make it more Idiomatic



# Constraints

What if?...

- Your codebase was 100x smaller?
- It needed 100x less RAM?
- It ran on a microchip?



# Pitfall:
# Avoiding making things worse



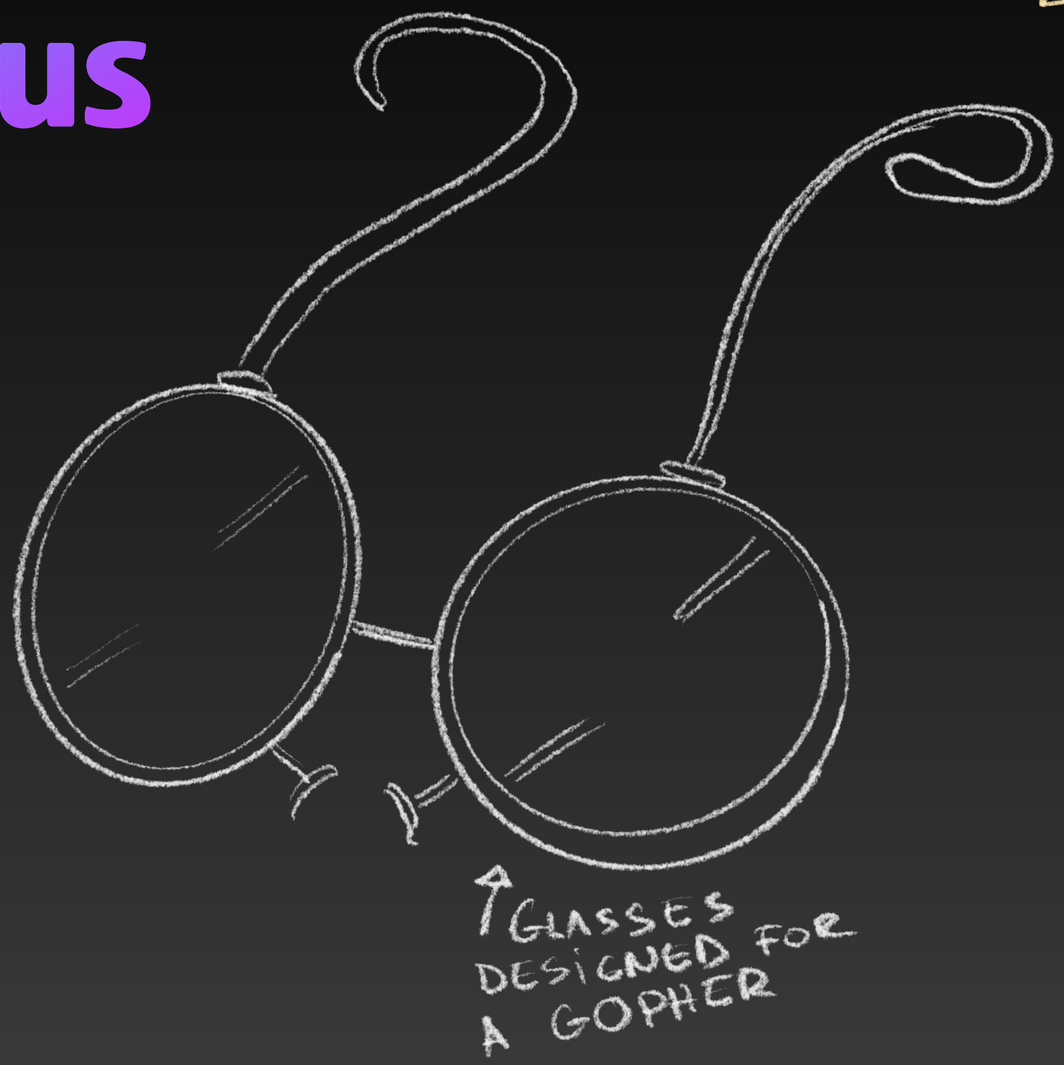
# Exercises:

1. SCAMPER the Aspects
2. Collect 100 Transformations
3. Find meaning of “15 Properties in Nature”
4. Transformation Drill (practice every single tf. on few ideas)
5. Transform the Transformations

# Part 4: Placing ideas into the world

* WIP

# Ideas need focus and clarity.




*Placing into the world*

# Understanding the context of an idea

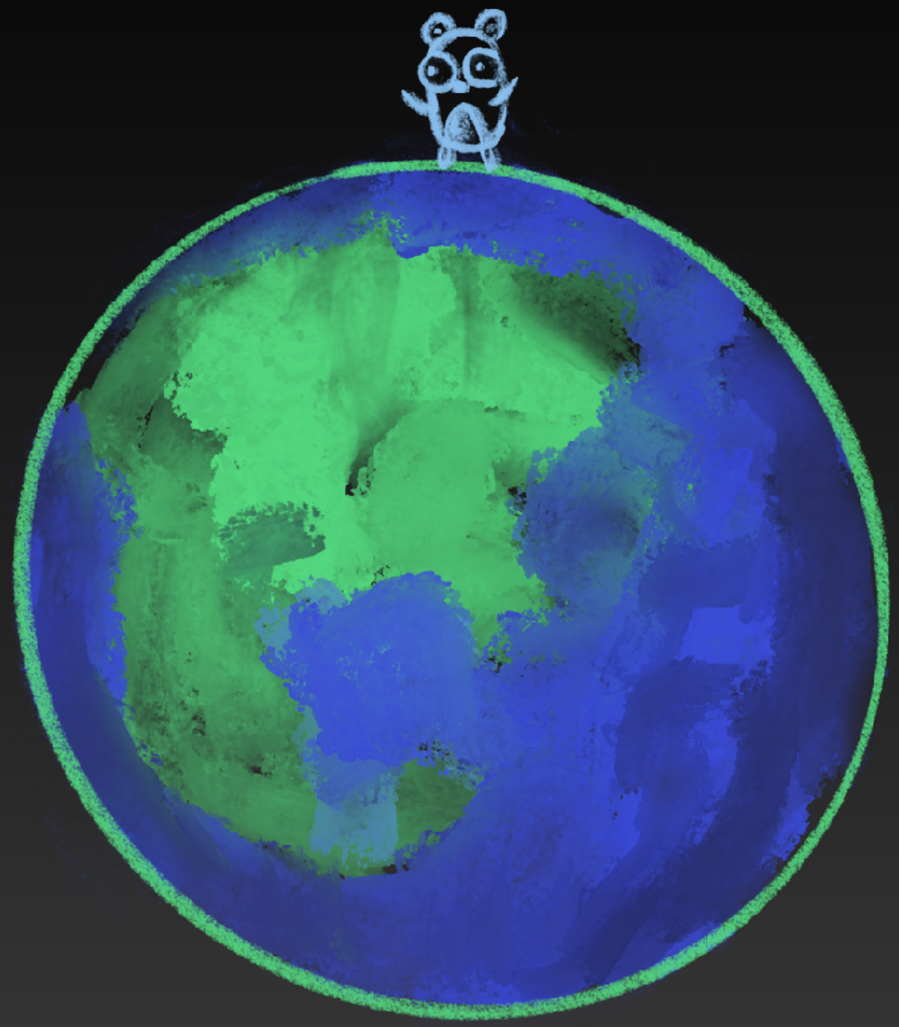
Who and what cares about our idea?

Why do they care?

What are the constraints imposed on the idea?

What other ideas are also involved?

How will the idea be realised?

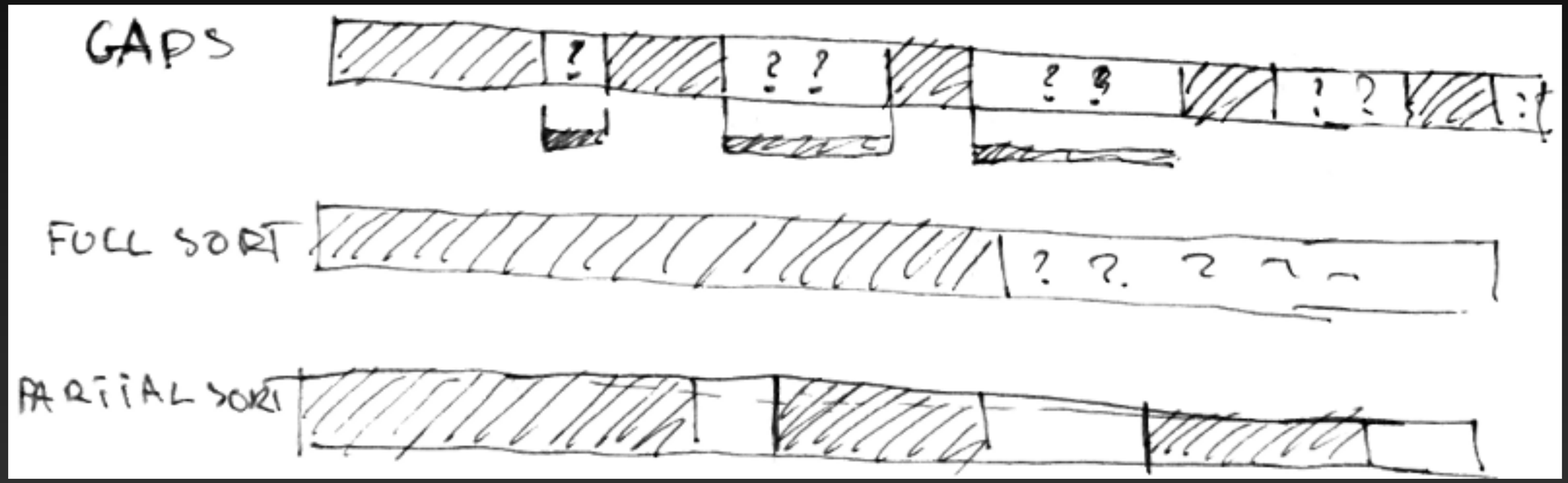




*for something more comprehensive see: iitoolkit.com*

# Pitfall: ***Confusing the solution for the problem***

## A breadth first search



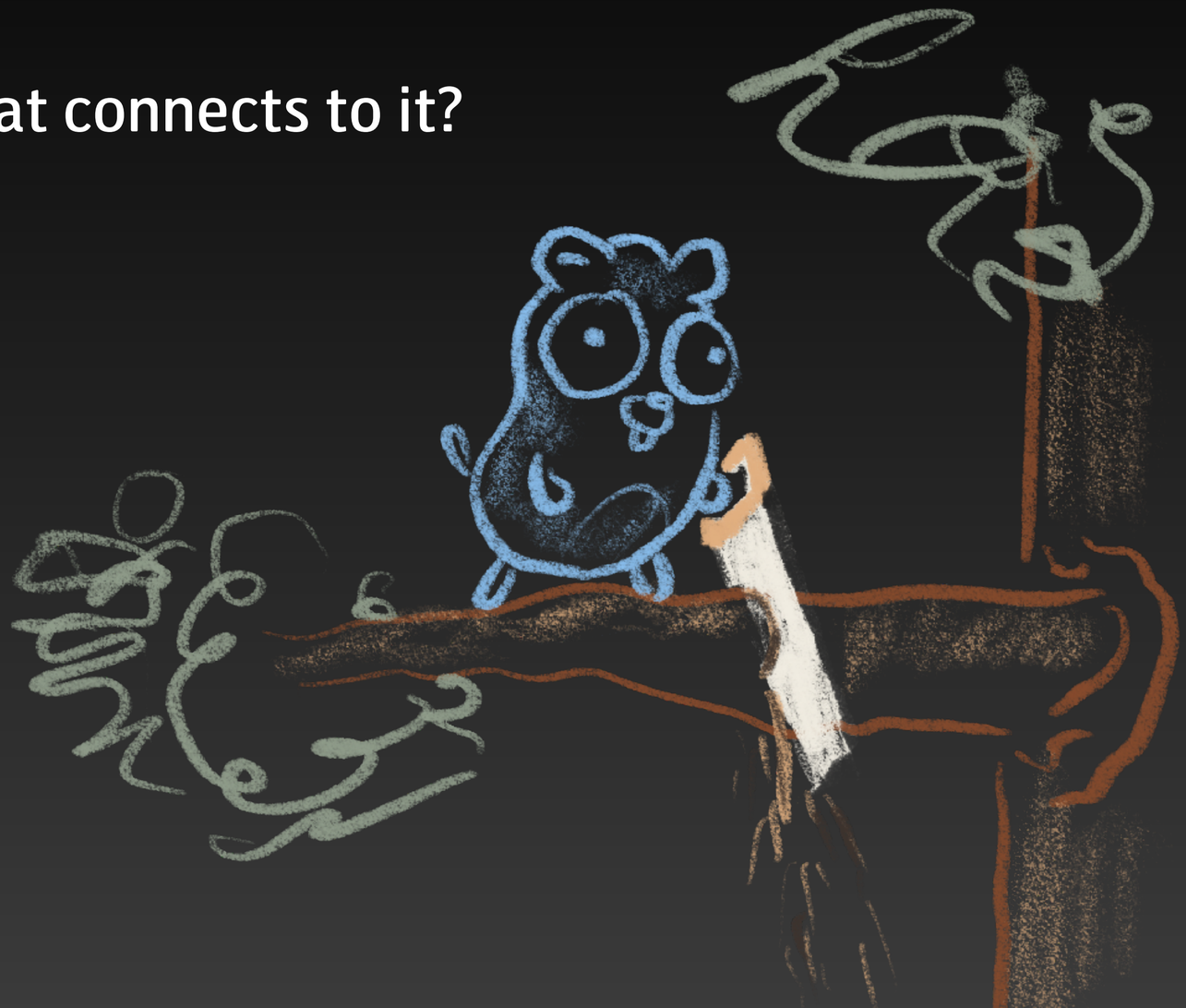

https://egonelbre.com/a-tale-of-bfs-going-parallel/

# Understanding the impact of an idea

How will it affect everything that connects to it?

What properties will it have?

How will it affect the world?



# Feeling the impact of an idea

What does it mean to you?

Why do you care?

How does it make you feel?



# "Mirror of the Self" Test

*by Christopher Alexander*

How well does it reflect me?
What is the truest picture of me?



*Placing into the world*

# Exercises:

1. Collect problem descriptions
2. Collect multiple frameworks for handling problems
3. Practice every framework on a given problem
4. Practice understanding the context.
5. Practice understanding the impact.
6. Practice understanding yourself.

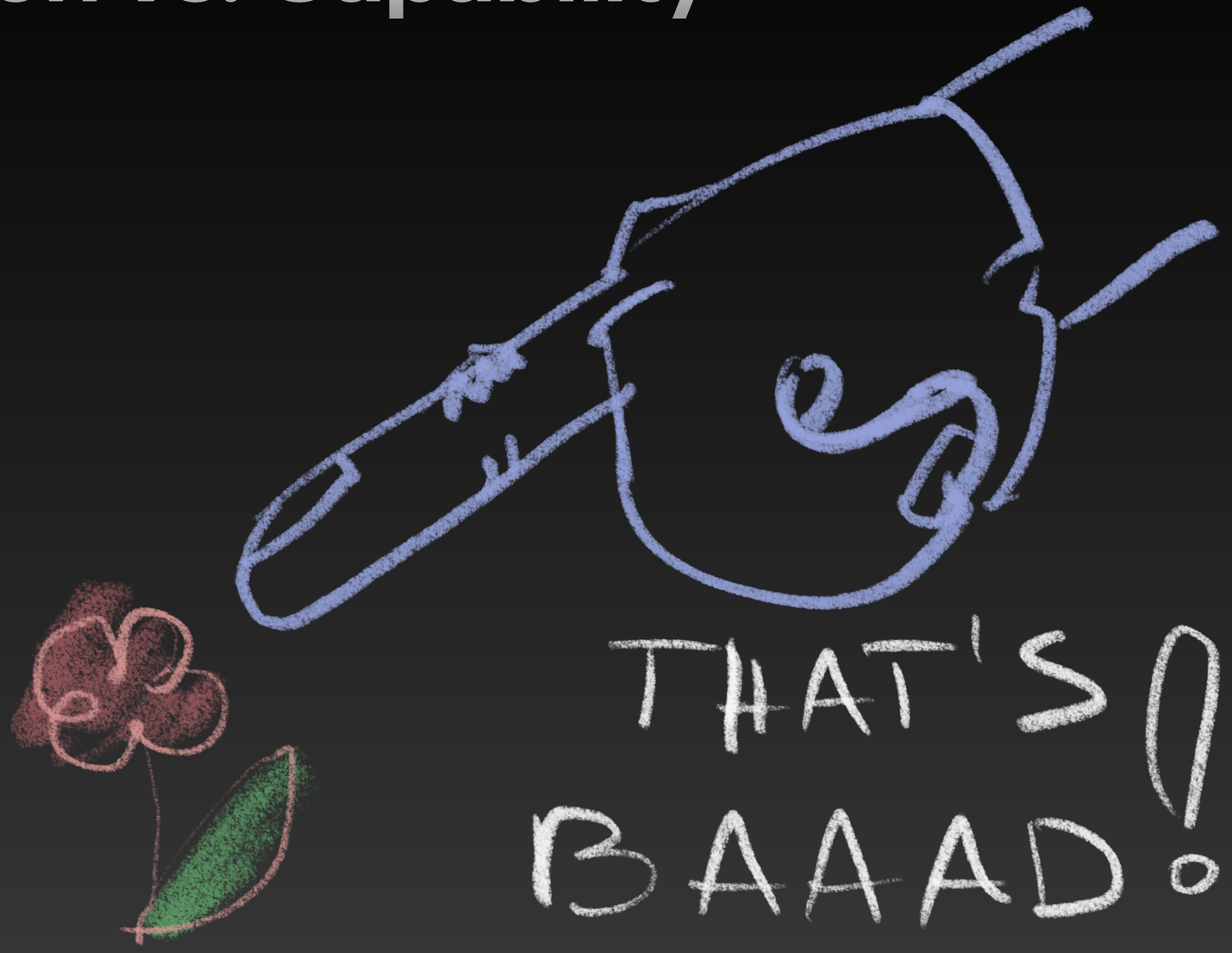
# Monsters of a Creative Adventure



# Noise



# Perception vs. Capability



# Muddy Middle



# Perfection Trap



# Freedom in Exploration



# It's fine to fail



# Conclusion

# Where Do Ideas Come From?

**Ideas come from other ideas**:

- *collect thousands of ideas,*
- *collect hundreds of categories,*
- *practice noticing ideas.*

**Ideas come from transforming**:

- *collect transformations,*
- *practice transforming ideas.*

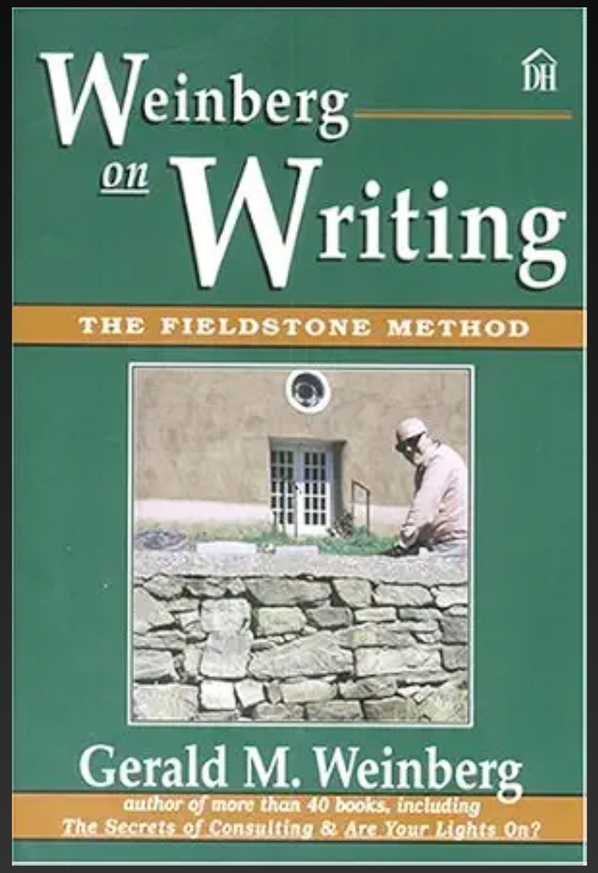
**Ideas come from understanding**:

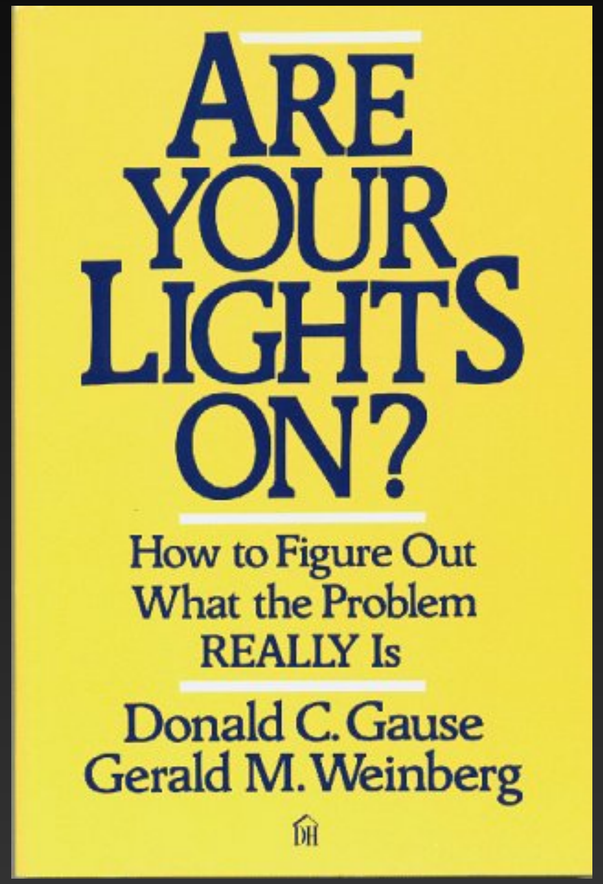
- *analyse your ideas,*
- *collect aspects,*
- *organise your ideas.*

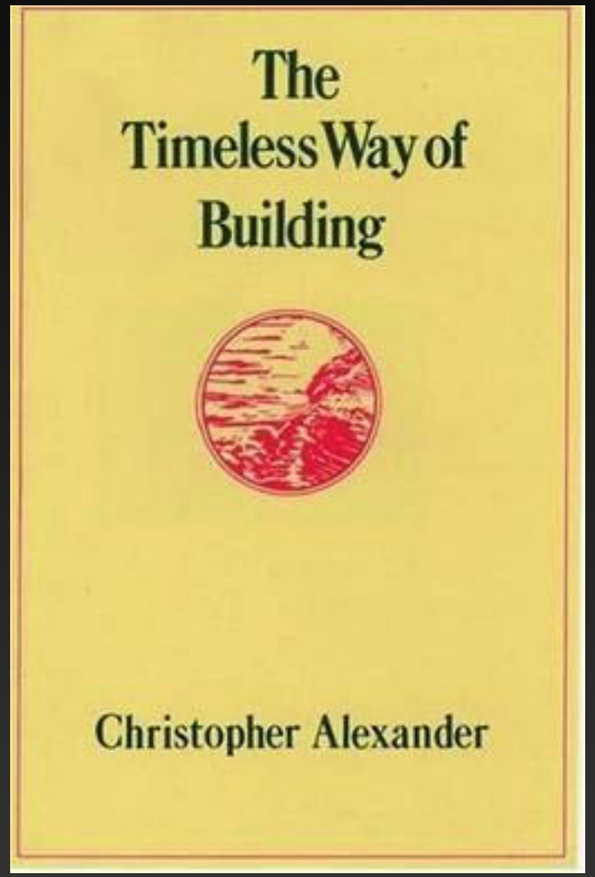
**Ideas need focus**:

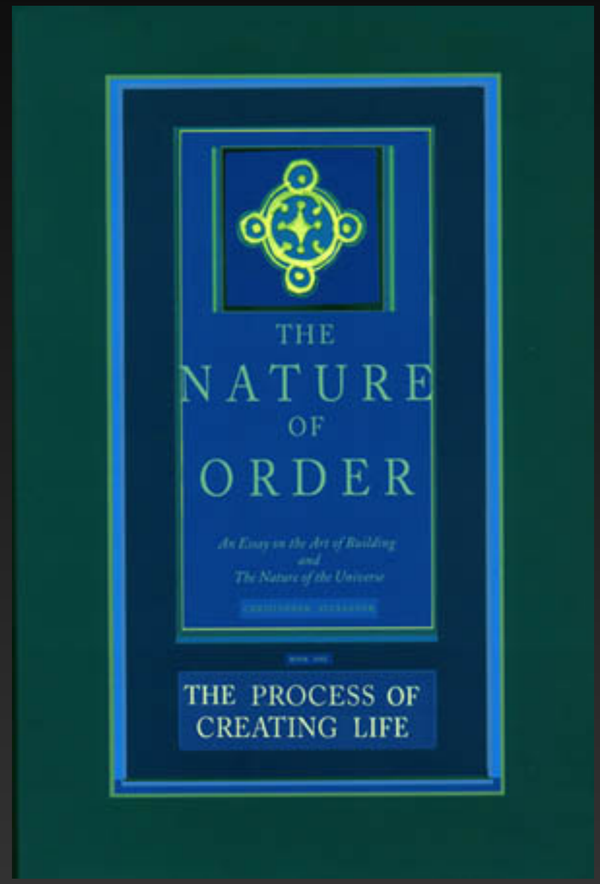
- *understand why you are doing something,*
- *understand how it fits into the world*
- *understand how it impacts the world*

# Further reading



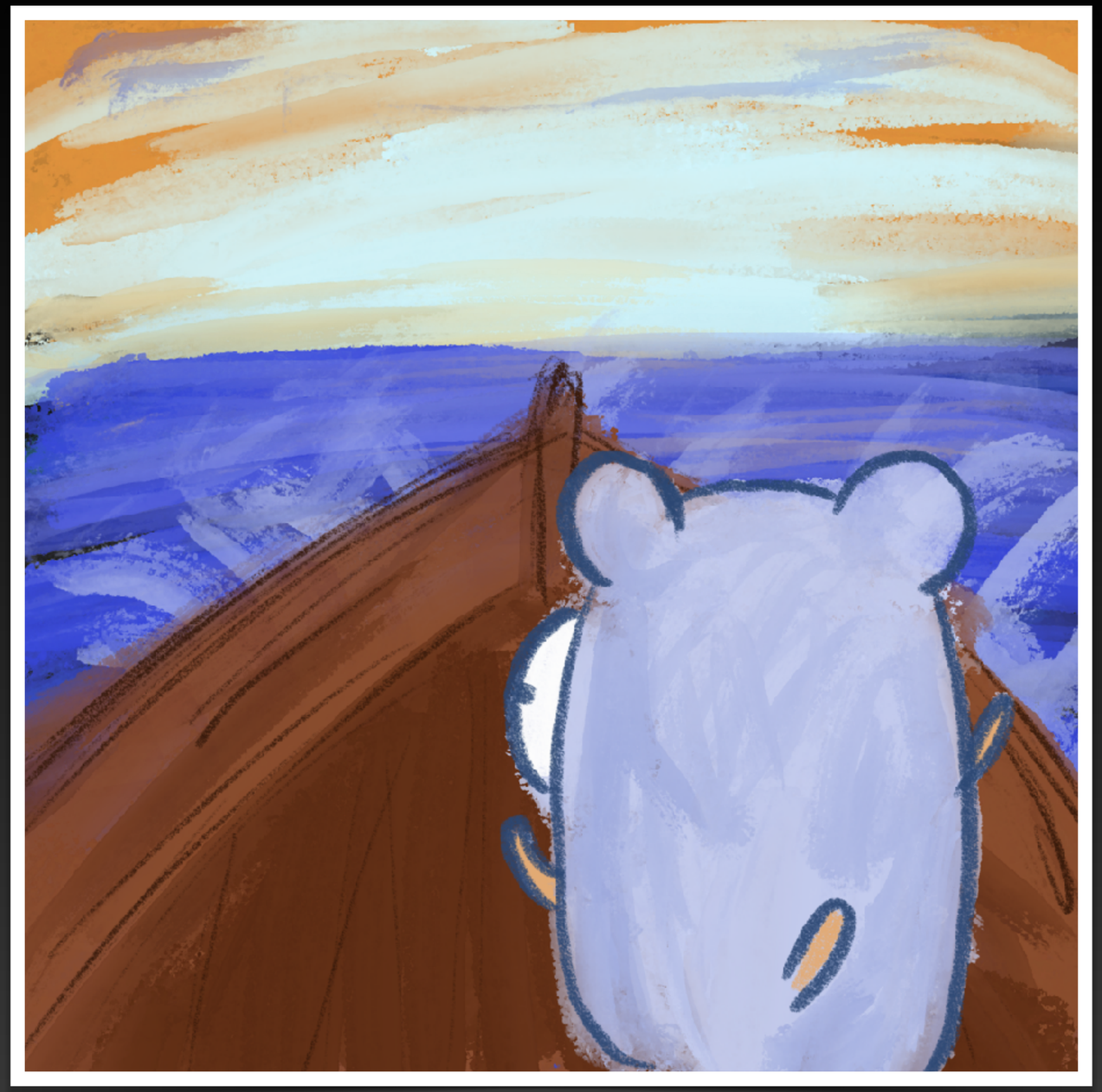




actually about collecting and developing ideas


NICE!

Egon Elbre
egonelbre.com



A Vast World
To Explore