Practical Odin for gamedevs

By Karl Zylinski – https://zylinski.se

- A programming language
- C-like
- High performance
- Low level
- Simple
- Comfortable, modern features: “low level with high level feeling”

Understanding the ODIN Programming Language

Karl Zylinski

Why did I get into Odin?

At

we programmed C. We enjoyed:

- Custom allocators
- Temporary allocators
- Tracking allocators
- Designated initializers
- Zero is initialized
- Cache friendly programming

I ran into and saw that it featured:

- Custom allocators
- Temporary allocators
- Tracking allocators
- Designated initializers
- Zero is initialized
- Cache friendly programming

Let’s go!

Simple vector maths

Using array programming!

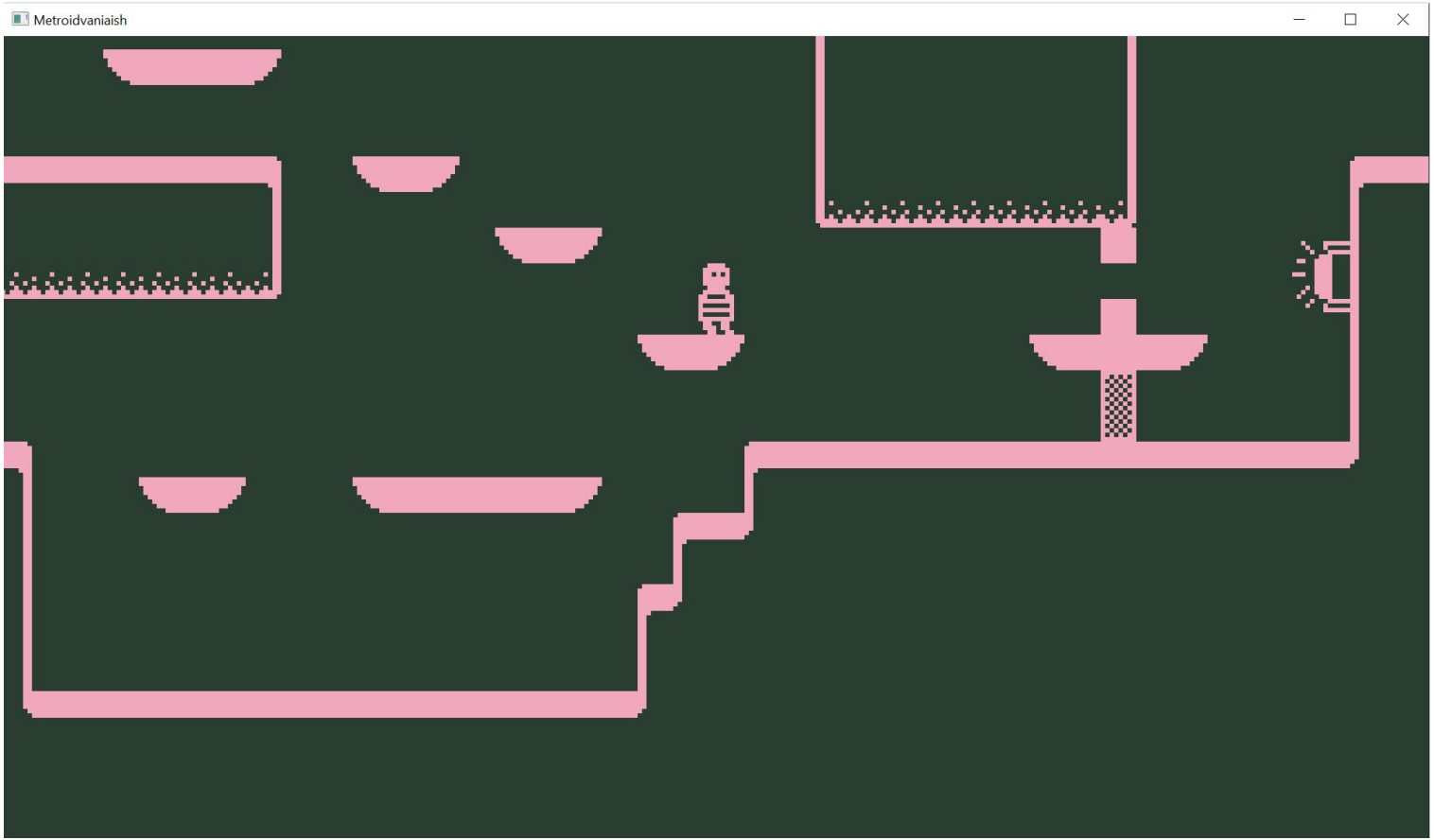
Don’t be scared of memory leaks

Track the leaks!

Making use of the temporary allocator

Enumerated arrays: Connect gameplay concepts

State machines using tagged unions



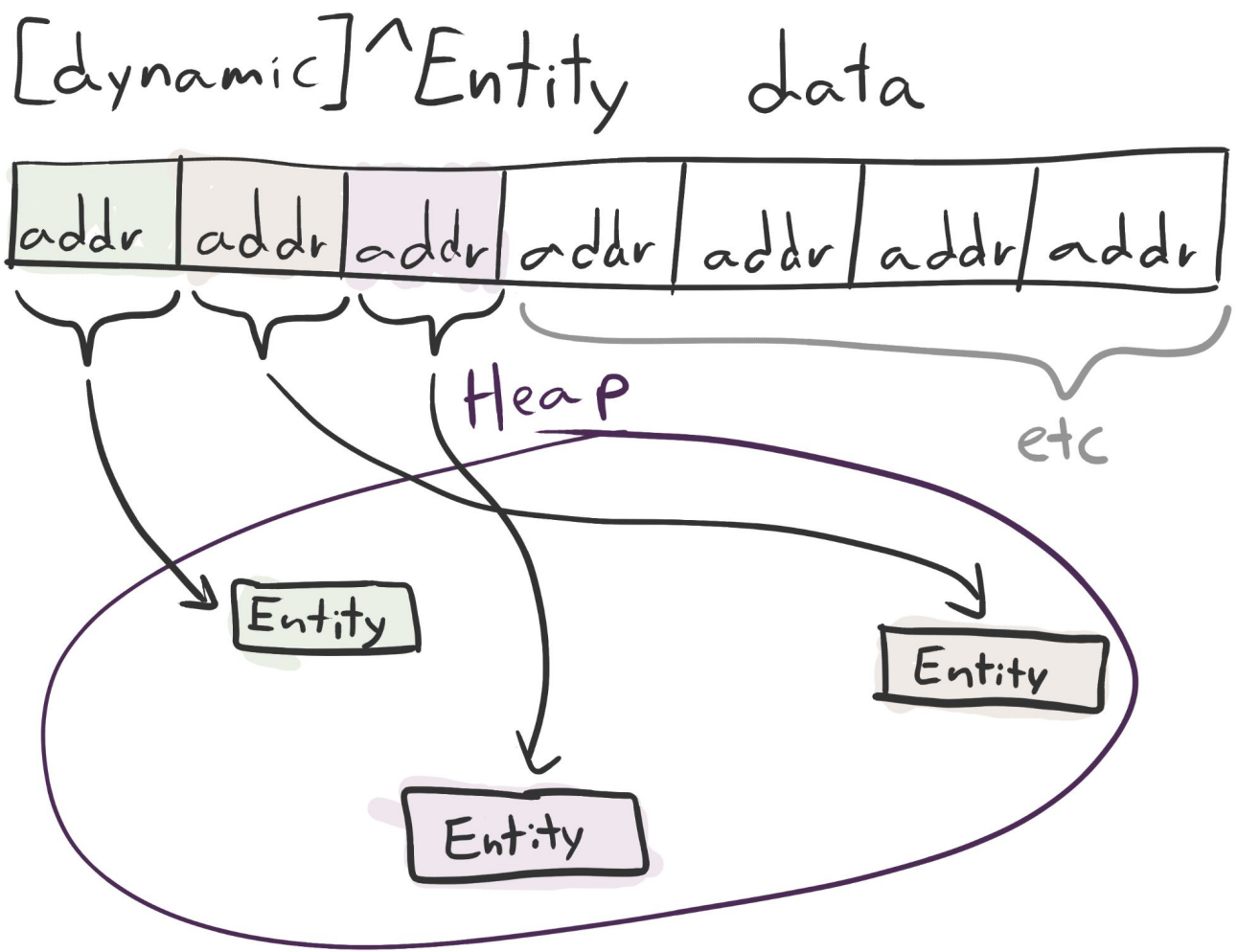
Entities: 4 approaches

Let’s look at some code..

STOP IT
YOU DON'T NEED ECS

Don’t separately allocate elements in big arrays

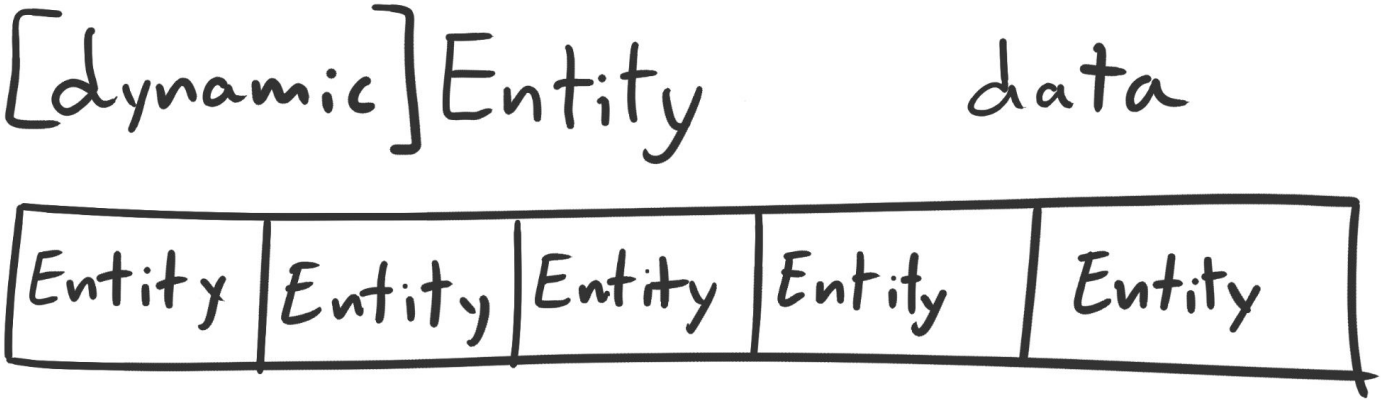
~~entities: [dynamic]^Entity~~



Understanding the ODIN Programming Language

Karl Zylinski

entities: [dynamic]Entity



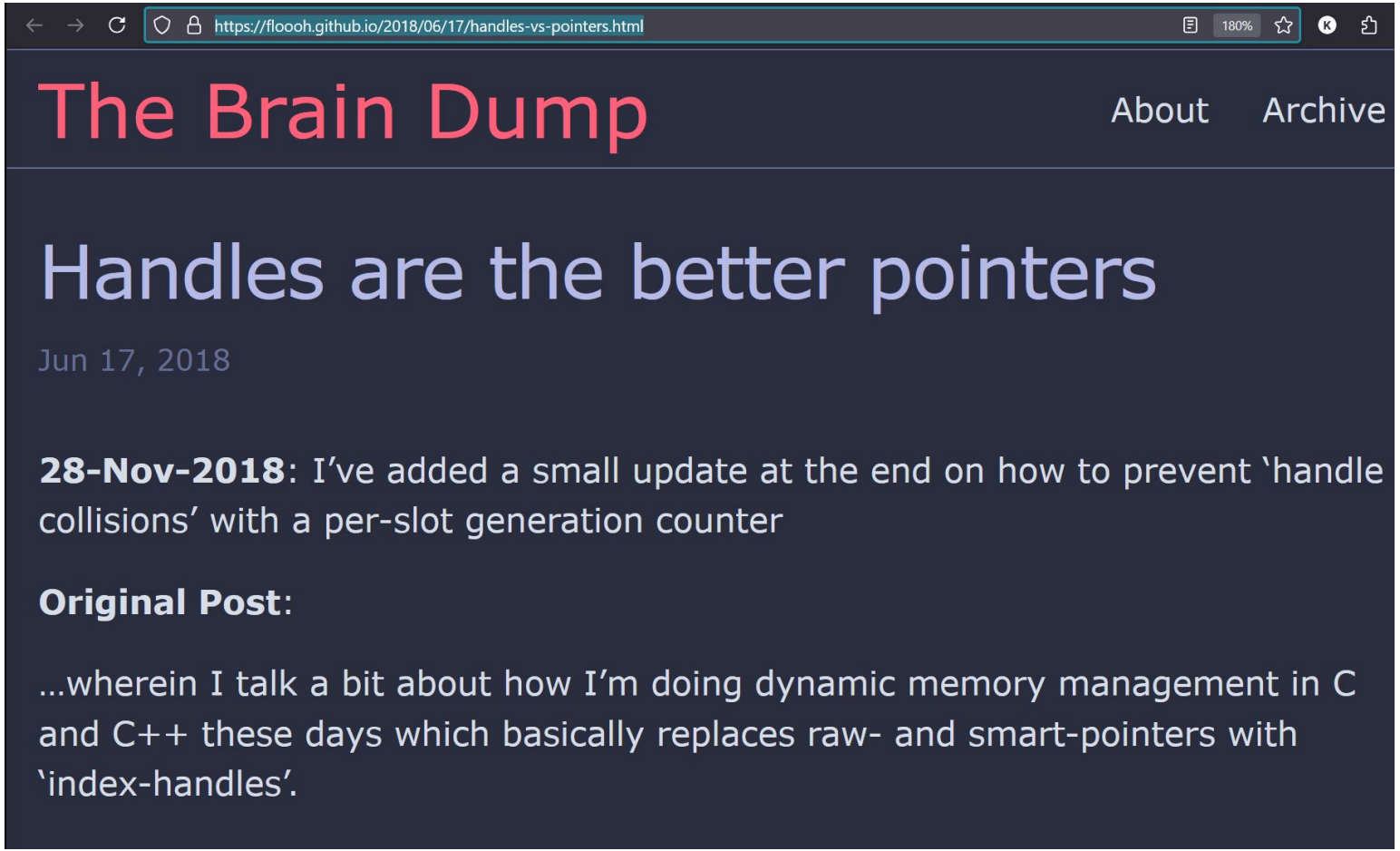
Wait a minute…

```
1 entities: [dynamic]Entity
2 append(&entities, Entity { bla bla })
3 entity_ptr := &entities[0]
4
5 for i in 0..<1000 {
6     // This will make entities grow!
7     append(&entities, Entity { bla bla })
8 }
9
10 // May crash
11 fmt.println(entity_ptr.some_field)
```

Not a reason to separately allocate!

Instead: Use a handle-based map

https://floooh.github.io/2018/06/17/handles-vs-pointers.html



https://github.com/karl-zylinski/odin-handle-map

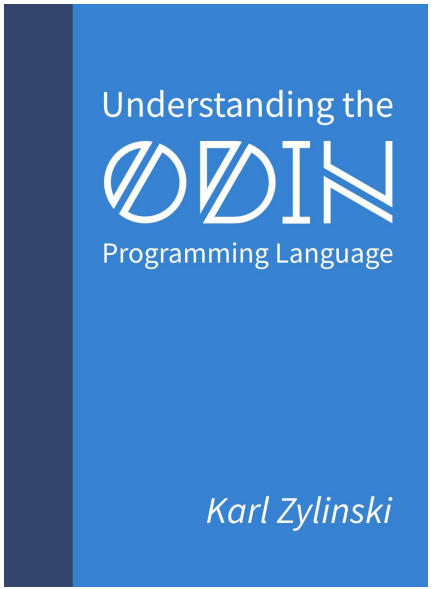
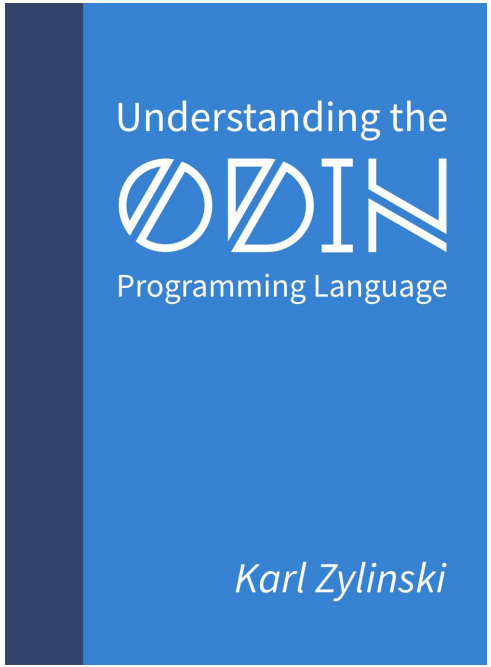
Thank you!

Learn Odin using my eBook “Understanding the Odin Programming Language”:
https://odinbook.com

My website and blog:
https://zylinski.se

These slides:
https://zylinski.se/fatshark.pdf

Don't use pointer parameters for the sake of optimization



“Prefer to pass slices”