The art of Programming
30 December, 2023

INSTRUCTOR: ME
It is just a tool

Beginner:

Different Programming Languages Creation:

- Trade-offs that each offered. languages that don't give you precise control over the hardware. because an audio workstation is one of those places where we're talking hard real-time. you have to be there on time.
- metaprogramming facilities. you mess around with strings and you have a lot of unwanted side effects.
- standard library ecosystem to access/extract different information. it depends on your OS, how easily accessible it is to expose the data through systems API calls.
- LLVM library, it's a unified framework for optimizing for generating optimized machine code. Your compiler reads your program that it's trying to compile, builds a data structure in memory, does semantic analysis, then the final step is to give some of that information to LLVM. which will then take care of selecting which exact instructions to use for the CPU that you're targeting. Since it's all bundled in. Then a C compiler that runs on LLVM. That's what allows us to build C.
- A compiler is just one step necessary to create a final executable. There's also a linking phase at the end. We had our own in-house linker. By having our own linker, we were able to reach a feature even faster than LLVM could. The point is that we did not consider LLVM the baseline. We were willing to get past LLVM and do some of that work ourselves. Now going forward, we plan to make LLVM a completely optional component. That means that we have our own implementation of some of what LLVM does.

- Working on LLVM, it's a humongous C++ project that takes forever to compile. It's messy.
- An array you know exactly how much memory it will need.

What tech to choose / Different Programming languages:

- ~/screenshot/audios/odin-language/
- odin - gaming industry or visual effects or tooling industry. mature all of the graphics apis already built into the library. metal natively, real-time more control and that's why fast. No need for direct graphics api calls.
- Zig can use any C library. Export function or import C functions. It's got the foreign import system. Because of the "application binary interface". Every language knows that. Zig toolchain rather than the language itself. "Focus on debugging your application rather than debugging your programming language knowledge".
    - Want to learn data structures and algorithms with zig.
- If you want something to be truly safe, rust is probably your best bet at the moment. odin gives you much greater control. Zig and Odin are kinda the same. only in doing things it's different and ecosystems are different.
- Zig is for compile time logic that executes at compile time. ex: calculate types.

Back-end engineering main components:

- Database
    - API design which basically How to assess that data through a HTTP server (end point).
    - Third-party services For the database
    - Infrastructure for the database

Investing in systems:

- I started as a systems engineer, moved to product and then eventually made my way to investing.
    - USCViterbi school of engineering, B.S. industrial & systems engineering to
    - Solutions Architect at Oracle on **database team** to
    - Product Lead for Cloud Firestore

- I always worked in databases and distributed systems. I love databases because they're the beating heart of a business.
- People want their ecosystem (OS) around data to be strong, apps to be built around that data. so microsoft, git, everybody is doing their api game.
- Writing your data on a distributed ledger instead of a database.