JOURNEY OF JOURNEY PROGRAMMING LANGUAGE WRITTEN IN RUST

By Tester (Atmaram Naik)

ABOUT SPEAKER

|Speaker|Blogger|Passionate Programmer|
|Tester | Opensource Contributor |

@atmnk9

Works at **Sedin Technologies** as **Test Manager**

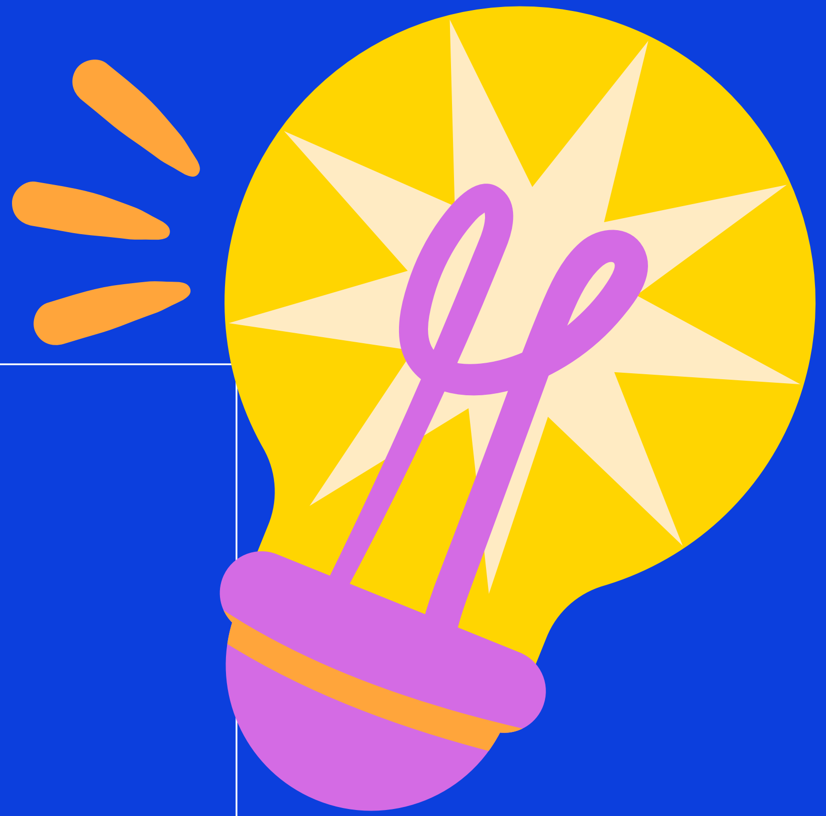
AGENDA

- The Idea
- Quick Demo
- How it started with Java
- Need for Different Language
- Rust As a Chosen Language
- Project Plan
- +/- Features
- Experiences with Rust
- Extended Demo
- Conclusion

THE IDEA

There should be something where I can quickly automate routine tasks as reusable commands

IDEA - ELABORATION



- I should be able to quickly create commands for multiple use
- The commands should be parametrised
- while running command if anything is not known it should be asked to invoker of command
- Commands can be created for following tasks
    - Test data creation activities
    - Creation of Mock servers

QUICK DEMO

Corr compiler:
Tool with programming ability

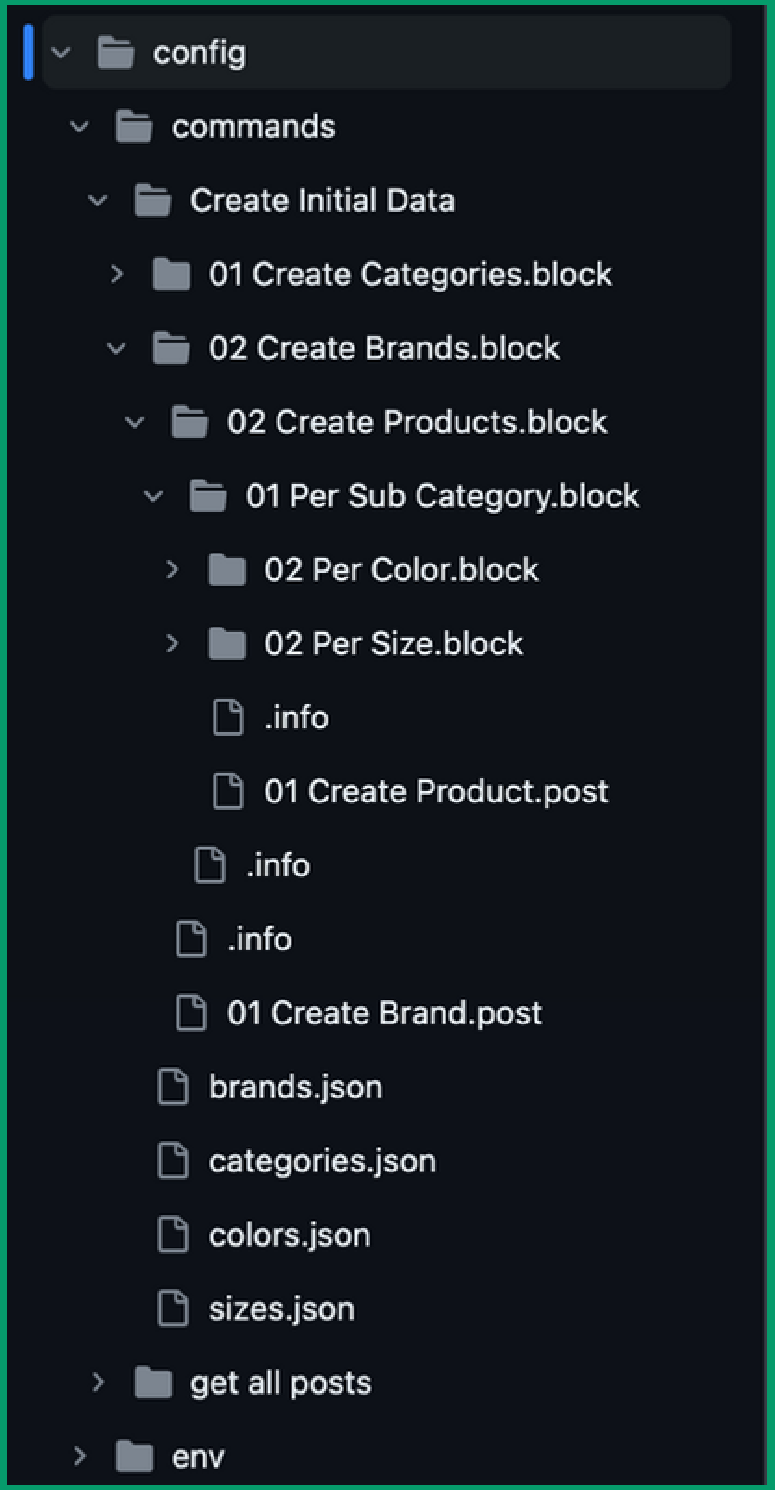
A Journey Programming Language
Opensource project

HOW IT STARTED

A lazy tester wanted to automate mundane activities

INITIAL PILOT WITH JAVA

- Entire tool was config driven
- For creating commands you needed to create folders and files in specific structure
- all control structures were config driven too
- Was highly designed on specific conventions that needed to be followed
- To run a tool you needed to have specific version of java and tool bundled as jar
- It was used for real project as productivity tool



```
config
  commands
    Create Initial Data
      01 Create Categories.block
      02 Create Brands.block
        02 Create Products.block
          01 Per Sub Category.block
            02 Per Color.block
            02 Per Size.block
            .info
            01 Create Product.post
          .info
        .info
        01 Create Brand.post
      brands.json
      categories.json
      colors.json
      sizes.json
    get all posts
  env
```

OVERALL FEEDBACK

- The idea is really good
- Tool like this simplifies lot of mundane activity
- Creating actual command is bit **difficult** and creator need to understand lot of convention followed
- The tool has **dependency** on specific version of java and user needs to **install java**
- The tool is **restrictive** in terms of possibilities that can be done with it.

Canva

NEED FOR DIFFERENT LANGUAGE

We need a language which can produce **native** tool

BASE LANGUAGE NEEDS

- Language should be able to produce cross platform executables
- Language should be memory safe since tool is going to be highly programmable
- Language should cater to high concurrency needs since tool can be further adopted for writing mock servers and load tests
- Ecosystem around language should have ready made libraries for creating cli tool
- Language should be systems programming language since tool may be further extended for lower level programmability
- Language should have low or no GC footprints since tool will be further extended for load testing
- Language should be type safe

EVALUATED LANGUAGES

C++

- Is Systems Programming Language
- Can produce native binaries
- Can support very high concurrency
- Has libraries for building cli tools
- Memory safety is not language feature and is a programmers job
- Has no GC footprint and memory management is programmers job

Rust

- Is Systems Programming Language
- Can produce native binaries
- Can support very high concurrency
- Has libraries for building cli tools
- Is Memory safe and its forced on programmer
- Has no GC footprint and rust forces memory management via borrow checker

Go

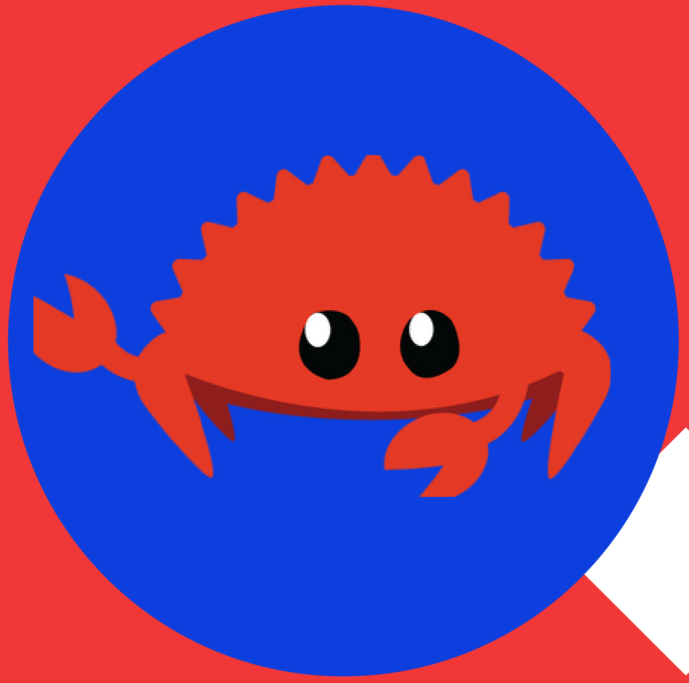
- Is Systems Programming Language
- Can produce native binaries
- Can support very high concurrency
- Has libraries for building cli tools
- Is Memory safe but its not forced on programmer
- Has Garbage collector which periodically scans heap for unused object

LOVELY RUST

Most **Loved** Programming Language

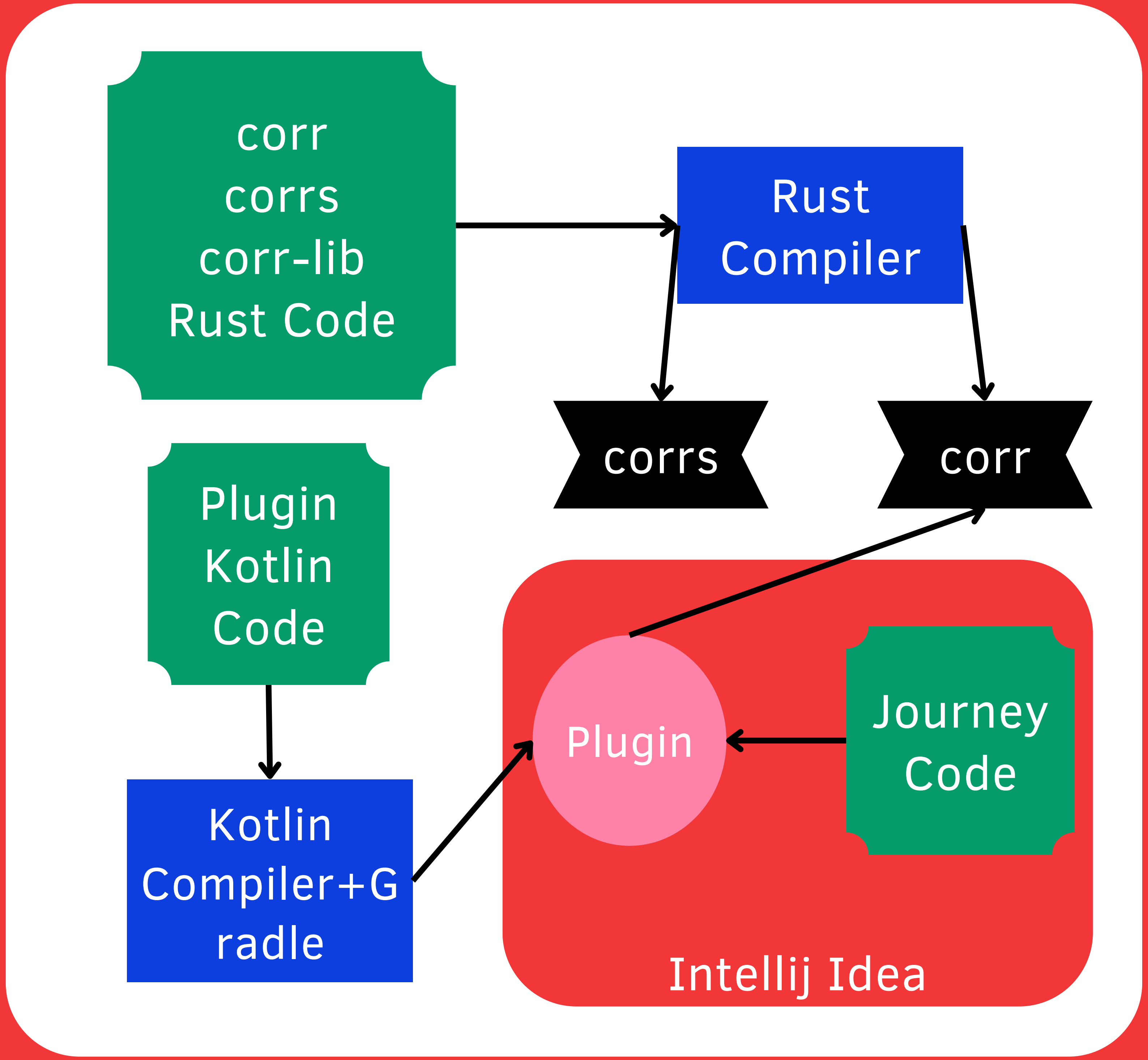
Type Saftey on steroids

"If your code compiles it will less likey have any issues"



PROJECT PLAN

- Design and Architecture
- Writing Unit, Integration and End To End Tests
- Creating Intellij Idea Plugin for adoption of Language
- Adding more features as and when required
- Evaluating existing Opensource Libraries for not reinventing wheel



DESIGN AND ARCHITECTURE

- Three crates and one plugin - Server, Compiler, Library, Intellij Plugin
  - Server (corrs)
    - A Websocket based server which can be interacted with
    - It can be integrated with any messaging applications
  - Compiler (corr)
    - Core Compiler that compiles project and creates distibutable package
  - Library
    - A library that has most meaty parsing and execution logic which can be reused in Server and compiler
  - Plugin
    - An Intellij plugin to provide intellisense arround syntax of language
- Unit, Integration and End to End Tests as safety net

+/- FEATURES

- Started As a tool for test data creation by invoking rest APIs
- Added feature of Service Virtualisation and mocking
- Added feature of Test Data creation directly in DB
- Added feature of Performing load tests utilising existing journey scripts
- Removed corrs (Server Component) which can be reintroduced since it had low adoption
- CI/CD Pipeline to automatically create platform specific installables like homebrew formula, Windows Installers and Debian Installers

EXPERIENCES WITH RUST

- Had intial typical struggles of fighting with rust compiler for code to compile
- Entire async ecosystem in rust is highly customizable.
- crates like tokio, async-traits, nom, clap, serde were highly used
- Needed to use nightly since many language features and libraries used required nightly features
- Some libraries were either not existent or were not in mature state so needed to create quick and dirty reusable libraries like async-rdbc etc
- Every time a code that was compiled was rewarding since it used to ensure something working was produced
- Writing unit, integration and e2e tests provided necessory safety net to modify features regulerly without breaking anything
- Got all answeres in community forums and on community discord servers when was I stuck

LIBRARIES USED

- nom - for creating parsers and AST
- tokio - As asynchronous rust runtime
- hyper - for low level servers
- serde and serde-json - for serialization and deserialization
- clap - for cli parser
- futures and future-utils - for some ready made features with async code
- async-trait - for lot of async code with traits
- async-recursion - for some recursive async code
- fake - for some fake data generation features
- mockito - for creating test doubles for tests
- And many more....

EXTENDED DEMO

CONCLUSIONS

- Overall writing a programming language in rust was very rewarding experience
- The tool created and programming language created is been used by my past employers and present for Service Virtualisation, Test Data Creation and Load Testing hence it had well adaptation as well
- The whole Rust Ecosystem is great place for Experiments, tools and projects like this
- The feared learning curve with Rust has minimal impact as the programmer settles with Rust it feels more rewarding to create great code than initial struggle of fighting with language
- No wonder why rust is called most loved programming language.
- I consider Journey Programming Language as my best work and feel really proud about creating something thats useful using Rust

THANKS

Freepik | Canva
For graphics and illustrations

Conf42
For opportunity to present

Rust Community
For awesome support and ecosystem