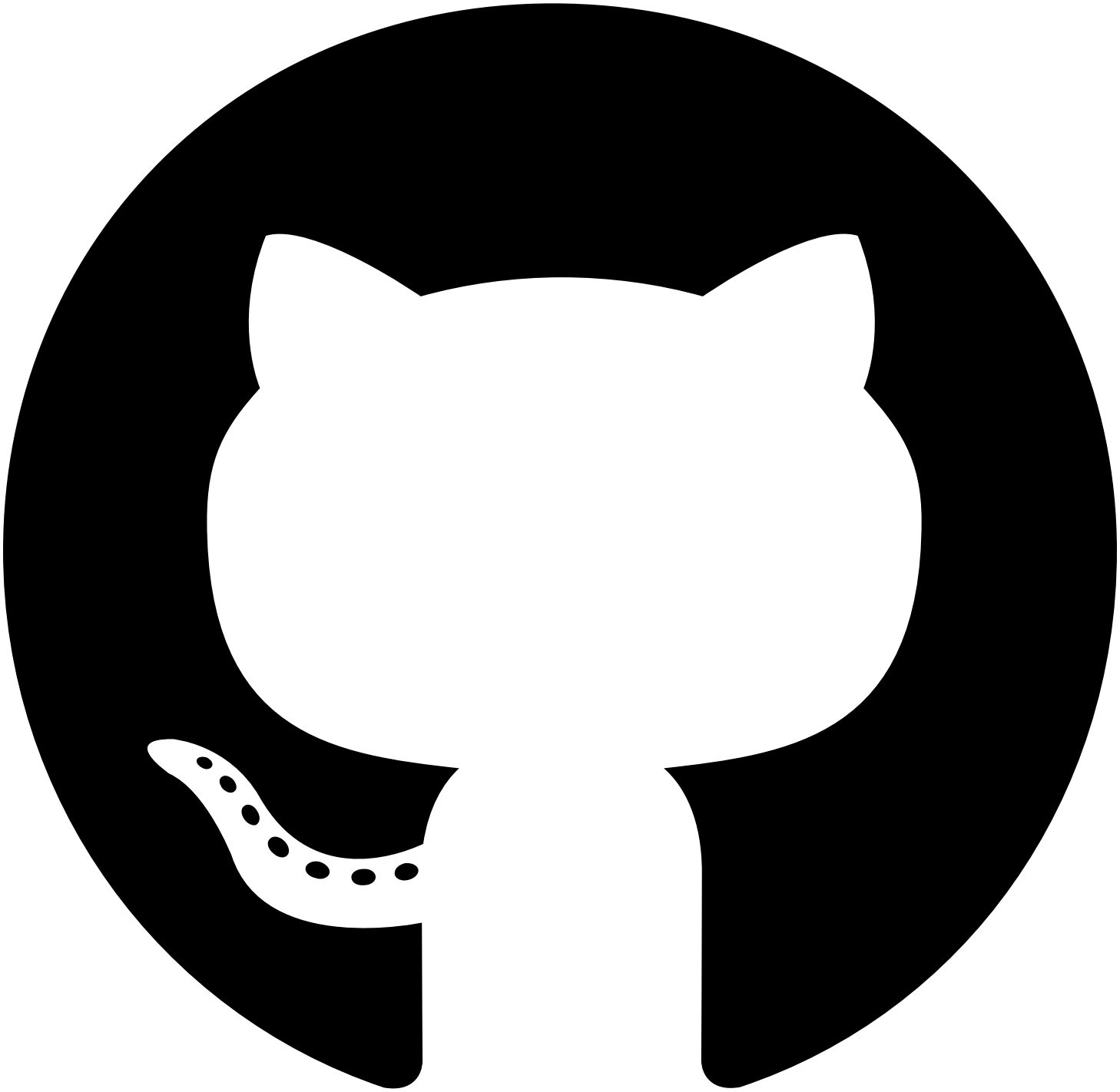
Rust Trends



28 - Rust in Action: 10 Project Ideas to Elevate Your Skills.
Posted on 2023-10-27

Hello, Rust Enthusiasts!

Welcome back to another edition of Rust Trends! You might notice that we're reaching out on a Friday this time, instead of our usual Wednesday. Why the change? Well, we want to set you up for a productive and educational weekend ahead!

In our second last issue, Rust 101: The Best Learning Resources Compiled, we explored the best resources to kickstart your journey with Rust. The positive reactions were overwhelming, and we are thrilled to see so many of you eager to learn. But what comes next after you have written your first "Hello, World!" program in Rust? The answer is simple: Projects, projects, projects! This is the best way to retain and build upon what you've learned so far.

There's no better method to solidify your understanding of a programming language than by building something from scratch. That is why this edition is dedicated to inspiring you with project ideas tailored for different skill levels. Whether you're a beginner looking for a small weekend project or an intermediate developer aiming for something more ambitious, we've got you covered.

So, let's turn those learning resources into practical experience and dive into the world of Rust projects! Get ready to make the most out of your weekend with Rust!

Rust in Action: 10 Project Ideas to Elevate Your Skills

1. grep command-line utility (Easy)

Create a simplified version of the Unix grep command-line utility. Your program should read a file and a search term (or regular expression) as input, then output all lines from the file that contain the search term. Rust Concepts Utilized:

- File I/O with the `std::fs` module for reading files
- Regular expressions using Rust's regex library for pattern matching
- String manipulation for searching and filtering lines
- Command-line argument parsing using `std::env`

-

This challenge is a fantastic way to get hands-on experience with text processing and pattern matching in Rust, making it an excellent project for those interested in systems programming.

2. URL shortener service (Medium)

Create your own URL shortener service. The application should take a long URL and return a shortened version, and when accessed, the shortened URL should redirect to the original long URL. This challenge will help you explore the following Rust concepts:

- Web server setup using frameworks like `Rocket` or `Actix`
- Data storage and retrieval, possibly using databases like SQLite or key-value stores like Redis
- String manipulation for generating short URLs
- HTTP request and response handling

This is a fantastic project for those looking to delve into web development with Rust and understand how to work with databases, web frameworks, and HTTP protocols.

3. Text-Based Adventure Game (Medium)

Create a simple text-based adventure game where the user can explore rooms, pick up items, and solve puzzles. This will help you get hands-on experience with:

- User input and output using Rust's `std::io`
- Structs and Enums for game states and items
- Control flow for game logic

4. Basic Web Scraper (Easy)

Build a web scraper that fetches and parses the content of a webpage, extracting specific information like headlines or links. This will involve:

- HTTP requests using libraries like `reqwest`
- HTML parsing using libraries like `scraper` or `select`
- String manipulation and regular expressions

5. Real-Time Chat Application (Medium)

Develop a real-time chat application where multiple users can join rooms and send messages to each other. This will allow you to explore:

- Networking with Rust's `std::net` or third-party libraries like `tokio`
- Multi-threading for handling multiple clients
- Data serialization and deserialization, possibly using formats like JSON

6. File Encryption and Decryption Utility (Medium)

Create a utility that can encrypt and decrypt files using a given key. This will involve:

- File I/O with `std::fs`
- Cryptographic algorithms using libraries like `ring` or `openssl`
- Command-line argument parsing with `std::env` or `clap`

7. Markdown to HTML Converter (Medium)

Build a tool that converts Markdown files to HTML. This will help you get hands-on experience with:

- Text parsing and manipulation
- File I/O with `std::fs`
- String interpolation and formatting

8. Simple HTTP Server (Medium)

Create a basic HTTP server that can serve static files and handle basic RESTful operations. This will involve:

- Networking with `std::net` or `hyper`
- File I/O for serving static files
- HTTP protocol understanding

9. Currency Converter with GUI (Medium)

Create a desktop application that allows users to convert between different currencies. The application should have a graphical user interface (GUI) where users can input the amount and select the currencies they want to convert between. Rust Concepts Utilized:

- GUI development with `Tauri`
- HTTP requests for fetching real-time currency exchange rates, possibly using `reqwest`
- State management to update the UI
- Event handling for user interactions like button clicks or dropdown selections

10. Image Processing Tool (Medium)

Create a tool that can apply basic image processing techniques like grayscale, blur, and edge detection on image files. This will involve:

- Image reading and writing with libraries like `image`
- Algorithms for image processing
- File I/O with `std::fs`

11. Bonus: Real-Time Object Detection System (Hard)

Create a real-time object detection system using Rust and machine learning libraries. The application should be able to process video streams and identify and label objects in real-time.

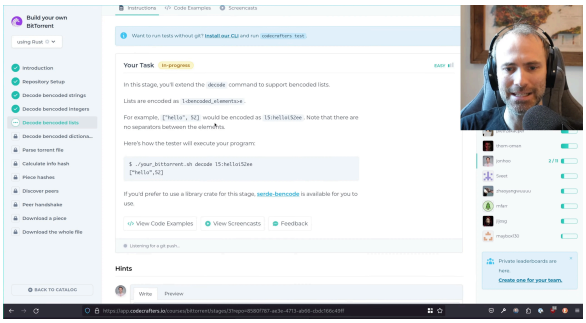
Implement features like bounding boxes around detected objects and real-time tracking. Rust Concepts Utilized:

- Interfacing with machine learning libraries through FFI (Foreign Function Interface) or using Rust-native libraries like `tch-rs` for PyTorch
- Video stream processing, possibly using libraries like `opencv` or `image`
- Multi-threading and concurrency for real-time processing
- Data serialization and deserialization for model input/output
- GUI for displaying the video stream and detection results, possibly using frameworks like `Tauri` or `GTK`

Each of these challenges is a great way to deepen your understanding of Rust's capabilities in different domains, from cryptography and text parsing to web development and image processing.

Interested in more ideas or want to share your project feel free to reply to this email.

Spotlight: Implementing BitTorrent with Jon Gjengset



In a recent live stream, Jon Gjengset took on an intriguing challenge: implementing a BitTorrent client in Rust from scratch and the 2nd part. The challenge, part of CodeCrafters' test-guided learning platform, serves as an excellent example of hands-on learning. Jon navigates through various aspects of BitTorrent and Rust, from decoding bencoded strings to peer handshakes, all while providing invaluable insights.

This is not a sponsored segment; Jon genuinely believes in this learning path and wants to share it with the Rust community. He even managed to troubleshoot and fix some off-by-one errors live, demonstrating the real-world challenges that come with coding. If you are interested in following along, you can find the final code changes on Jon's GitHub.

Whether you're a Rust newbie or a seasoned developer, this stream is a treasure trove of learning and inspiration. And if you decide to take up the challenge yourself, do not forget to use the referral link to sign up for CodeCrafters!

Snippets

- MinDNS: Minimal DNS server written in Rust
- Eureka: CLI idea notepad made in Rust

We are thrilled to have you as part of our growing community of Rust enthusiasts! If you found value in this newsletter, don't keep it to yourself — share it with your network and let's grow the Rust community together.

👉 **Take Action Now:**

- **Share:** Forward this email to share this newsletter with your colleagues and friends.
- **Engage:** Have thoughts or questions? Reply to this email.
- **Subscribe:** Not a subscriber yet? Click here to never miss an update from Rust Trends.

🔗 **Stay Connected:**

Feel free to connect with our editor, Bob Peters, on LinkedIn for more Rust insights and updates.

Sign up for our Newsletter