
Javascript Runtime

Section -3

Done By,
Sahana Lakshmipathy,
Naresh M,
(Dune Dev Squad)

Objectives

- The primary goal of this project was to enhance our custom JavaScript runtime by building a more engaging logging system.
 - We aimed to create a console debugging function that not only logs messages but does so with added timestamps and apt messages to make debugging more insightful.
 - The secondary objective was to deepen our understanding of how JavaScript runtimes work by customizing it with our own functionalities like `console.sarcasm()`.
-

Methodology

Custom Runtime Implementation:

- We began by setting up a custom JavaScript runtime using Rust and Deno's V8 engine.
- We also had to redefine the console object by adding new functionalities such as `console.sarcasm()` for more expressive logging.

Enhancing `run_js` Function:

- We modified the `run_js` function to load and execute a custom `runtime.js` file.
- This file defines our new console functions that include both standard logging with `console.log` and a sarcastic alternative with `console.sarcasm()`.

Testing and Iteration:

- To verify the effectiveness of our custom logging functions, we created and ran a test JavaScript file (`example.js`) with various logging scenarios.
 - We iterated through multiple versions, improving the console output by adding timestamps and refining the sarcastic message generation.
-

Tech Stack

- Rust
 - Deno's V8 Engine
 - JavaScript (Custom)
 - VS Code
-

Planning

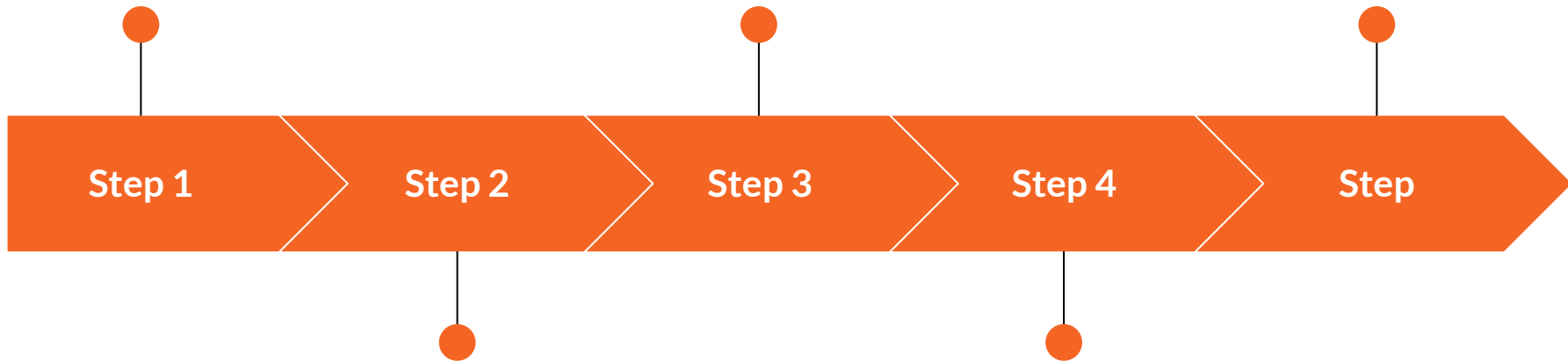
Defined the scope of the custom logging system.
Identified key features such as adding timestamps and sarcasm to logs adding color coding to each functionality.

Testing

Built a sample JavaScript file (example.js) to test the new logging functionalities. Checked all the functions for every added function

Final Review

As a team we conducted final tests and documented the performance and usage of the custom runtime.




Development

Created the `run_js` function in Rust to load and execute a JavaScript runtime (runtime.js).
Wrote the runtime.js file to redefine the console object, adding the `console.sarcasm()` function.

Debugging

Fine-tuned the runtime to handle edge cases in logging and ensured consistency in output formatting.
Ensured that timestamps are correctly displayed and sarcasm messages are varied.

Project Output

```
myjs >  example.js
1 console.log("Logger Message")
2 console.sarcasm("my custom function's message")
3 console.debug("This is a debug message.")
4 console.sarcasm("This is a sarcastic message.")
```

```
PS C:\Users\SEC\Desktop\js_runtime> cd myjs\
PS C:\Users\SEC\Desktop\js_runtime\myjs> cargo run
    Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.25s
    Running `target\debug\myjs.exe`
[out]: "Logger Message"
[19:18:34][message]: "my custom function's message"
You must be a real expert...
[19:18:34][message]: "This is a sarcastic message."
Congratulations, you broke the code
PS C:\Users\SEC\Desktop\js_runtime\myjs> |
```