
Visual Debugging on Replit: A Step Towards Intuitive Learning in Coding

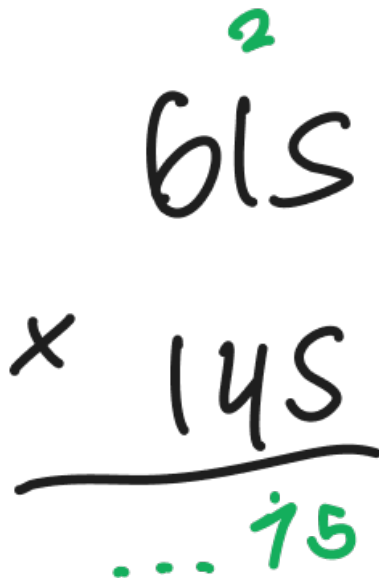
Entry for Replit Design Summer Internship '24. Transforming abstract concepts into visual narratives to revolutionize the coding experience for learners and professionals alike.



A handwritten equation $2 + 2 = 4$ in black ink. The number 4 is written in green ink.

Seems easy enough

As we sit down to solve a math problem, we often reach for a pen and paper, breaking down complex calculations step by step. It's a process that not only leads us to the correct answer but also deepens our understanding of the underlying principles. This approach isn't just for math; it's vital for coding too.



A handwritten multiplication problem. The number 615 is written in black ink, with a green 2 written above it. Below it, 145 is written in black ink. A horizontal line is drawn below 145. Below the line, the text "... 75" is written in green ink.

Imagine doing this slightly-more-complicated problem in code. Column by column.

I'm Carl, a computer science student at Minerva University, and I've been thinking about how we learn and engage with code on platforms like Replit. We're in an era where visual learning isn't just preferred; it's essential, especially for complex concepts like recursion or compound interest calculations in programming. That's why I propose a visual debugger for Replit, a tool that breaks down code execution in a step-by-step manner similar to solving a math problem.

Here's me basically saying what's being said in this article.

Imagine this: you're working through your code and with each line, a visual cue guides you to the next, showing you where variables are stored, how functions are executed, and where your logic might be tripping up. It's about making the invisible, visible.

This feature isn't entirely new—it's inspired by tools like Python Tutor and has been implemented on educational websites. But it's not yet a staple on Replit, a platform that's becoming a household name in the coding education space. This debugger could be transformative, especially for students and educators, making it easier to demystify coding concepts. And let's not overlook seasoned developers; such a tool could make debugging less of a chore and more of a learning experience.

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

```
Python 3.6
known limitations
1 # Define the principal amount (initial investment)
2 principal = 10000
3
4 # Define the number of times interest is compounded per year
5 compounding_frequency = 12
6
7 # Define the annual interest rate as a decimal
8 annual_interest_rate = 0.08
9
10 # Ask the user for the number of years to calculate compound interest for
11 years = int(input("Compound for how many years? "))
12
13 # Calculate the intermediate values
14 rate_per_period = annual_interest_rate / compounding_frequency
15 total_periods = compounding_frequency * years
16
17 # Calculate the final amount using the compound interest formula
18 final_amount = principal * ((1 + rate_per_period) ** total_periods)
19
20 # Display the result
21 print("The final amount after", years, "years is", final_amount)
```

Print output (drag lower right corner to resize)
Compound for how many years? 19

Frames Objects

Global frame

principal	10000
compounding_frequency	12
annual_interest_rate	0.08
years	19
rate_per_period	0.00666667
total_periods	228
final_amount	45492.197

⇒ line that just executed
⇒ next line to execute

Edit this code

105-1731-18/9 (2)

My annotations on Python Tutor

Yes, there will be challenges. Scalability and the complexity of integrating this into Replit's existing framework are not trivial. Yet, I believe the benefits—for learning, for debugging, for cognitive clarity—outweigh these hurdles. After all, isn't the essence of coding solving problems?

Success could be measured in several ways: engagement metrics, user feedback scores, and perhaps most tellingly, the adoption rate within educational institutions that use Replit.

With my background in product design, including translating designs into functional React components, I am keenly aware of the impact well-executed features can have on user experience. A visual debugger on Replit isn't just another feature; it's a step towards redefining how we learn to code in a visual world.

Carl Kho—Product Designer | Portfolio 2023

I'm a student at Minerva University and a San Francisco-based UI/UX designer/developer with a knack for creating...www.carlkho.com

By [Carl Kho](#) on [December 20, 2023](#).

[Canonical link](#)

Exported from [Medium](#) on October 31, 2025.