Praxly 2.0: Refactoring an Online IDE for Pseudocode Education

Victor Del Carpio · Chris Mayfield, PhD
College of Integrated Science & Engineering, Department of Computer Science

Introduction

Praxly 2.0 is a web-based educational IDE that simplifies learning programming logic and syntax. It offers real-time translation between Java, Python, and AP CSP Pseudocode, allowing students to master cross-language concepts through a unified debugging and execution interface.

Motivation

- **Architectural Refactoring**: Transitioned from legacy JavaScript to a TypeScript and React ecosystem.
- **Enhanced Maintainability**: Improved system-wide type safety and modularity for sustainable development.
- **Scalable Foundation**: Built upon original translation concepts to unlock expanded educational potential.

New Features

Syntax Highlighting
Lezer-powered semantic coloring for cross-language readability

IntelliSense / Autocomplete
Contextual logic suggestions facilitating error-free code entry

Source-Mapped Debugger
Synchronized statement stepping with source-to-target highlighting

A complete tech-stack migration and feature expansion of an open-source pseudocode IDE, enabling broader classroom adoption.

PRAXIS — SOURCE

```
int newScore ( int diceOne, int diceTwo, int oldScore )
  if ( diceOne == diceTwo )
    return 0
  else
    if ( ( diceOne == 6 ) or ( diceTwo == 6 ) )
      return oldScore
    else
      return oldScore + diceOne + diceTwo
    end if
  end if
end newScore
```

CSP VIEW

```
PROCEDURE newScore (diceOne, diceTwo, oldScore)
{
  IF (diceOne = diceTwo)
  {
    RETURN 0
  }
  ELSE IF (diceOne = 6 OR diceTwo = 6)
  {
    RETURN oldScore
  }
  ELSE
  {
    RETURN oldScore + diceOne + diceTwo
  }
}
```

Figure 1. Praxly2 running in-browser — Praxis Pseudocode (left) translated AP Computer Science Principles Psuedocode (right)

PYTHON — SOURCE

```
x = input()

def check(val):
  if val == "Hello, world!":
    return True
  else:
    return False

result = check(x)
print(result)

```

JAVA VIEW

```
import java.util.Scanner;

public class Main {
  public static boolean check(String val) {
    if (val.equals("Hello, world!")) {
      return true;
    } else {
      return false;
    }
  }

  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    String x = scanner.nextLine();
    boolean result = check(x);
    System.out.println(result);
  }
}
```

Figure 2. Praxly2 translating and inferencing type from Python (left) to Java (right)

Challenges

- **Unified AST Design**: Creating a single internal representation that captures the semantic nuances of all supported languages
- **Synchronous State Tracking**: Mapping interpreter steps to editor character indices for precise debugger highlighting
- **Grammar Reconciliation**: Merging disparate language grammars into a cohesive parser without losing syntax-specific features

Future Plans

- **Memory Visualization (MemDia)**: Generate diagrams in real-time to show variables, objects, arrays, and references.
- **AI Panel**: An integrated side panel that uses natural language processing to explain complex code and logic in plain English

Acknowledgement

The primary author gratefully acknowledges the support of the Department of Computer Science at JMU. This work was conducted as part of an independent study under faculty supervision during the Spring 2026 semester.