JOHANNES KEPLER
UNIVERSITY LINZ

Master's Thesis

Web-Based Bytecode Interpreter Visualization

Student: Tobias Herber, BSc

Advisor: Dipl.-Ing. Dr. Markus Weninger, BSc

Start date: September 2024

Dipl.-Ing. Dr. Markus Weninger, Bsc
Institute for System Software
T +43-732-2468-4361
markus.weninger@jku.at

The "Compiler Construction" course at Johannes Kepler University Linz, Austria, presents students with the challenging task of developing a complete compiler for MicroJava, a simplified variant of Java. This comprehensive project encompasses various compiler phases, including scanning, parsing, symbol table management, and bytecode generation. While MicroJava's reduced feature set (lacking inheritance and supporting only a limited set of built-in types) simplifies the task somewhat, students still face significant complexity in understanding and implementing each compiler component.

To support student learning and enhance their grasp of compiler theory and practice, the course has already incorporated interactive visualizations for the scanner, parser, and operand generation during code generation. These tools have proven invaluable in helping students understand how their compiler implementations process source code through different phases. However, a critical gap remains in visualizing the final stage of compilation: bytecode interpretation.

The bytecode interpreter, a stack-based virtual machine, interacts with four distinct memory regions: the data section for global and static data, the heap for objects and arrays, the function stack for local variables, and the expression stack for temporary data storage during computations. The intricate dance of bytecode instructions across these memory areas—loading values, performing operations, and storing results—often remains opaque to students, making it challenging to identify and rectify errors in their code generation phase.

Goal:
The primary objective of this master's thesis is to develop a web-based visualization tool that illuminates the inner workings of the MicroJava bytecode interpreter. This interactive visualization will serve as a powerful educational aid, allowing students to observe and understand the step-by-step execution of bytecode instructions and their effects on the interpreter's memory regions.

Key features of the visualization tool should include:

1. Step-through functionality: Enable students to navigate through bytecode instructions one at a time, providing a granular view of the interpretation process.

2. Animated data flow: Visually represent the movement of data between memory regions, using color-coding and animation to illustrate how each instruction affects the interpreter's state.

3. Error highlighting: When discrepancies arise between a student's generated bytecode and the expected output, the tool should clearly indicate these differences and demonstrate how the correct bytecode would behave.

JOHANNES KEPLER
UNIVERSITY LINZ

4. Instructor mode: Design the tool to be suitable for in-class demonstrations, allowing lecturers to replace traditional slide-based teaching with dynamic, interactive examples.

The development process should involve a thorough evaluation of existing teaching materials, particularly how interpreter internals and bytecode instructions are currently visualized in course slides. This analysis will inform the design of the new tool, ensuring it aligns with and enhances the current curriculum.

By creating this visualization tool, the thesis aims to bridge the gap between theoretical understanding and practical implementation in compiler construction. It will empower students to debug their code more effectively, deepen their comprehension of bytecode interpretation, and ultimately enhance their overall learning experience in the course.

The student undertaking this thesis will need to demonstrate proficiency in web development technologies, a strong understanding of compiler theory and bytecode interpretation, and the ability to create intuitive, educational user interfaces.

Modalities:
The progress of the project should be discussed at least every three to four weeks with the advisor. A time schedule and a milestone plan must be set up within the first two weeks and discussed with the advisor(s). It should be continuously refined and monitored to make sure that the thesis will be completed in time. The final version of the thesis is expected to be finished before 31.08.2025.