Code Visualization

Simplifying debugging through the visual representation of data structures and their behavior

People Involved

Students

- Curtice Gough
- Catherine DiResta
- Joshua Hartzfeld

Faculty

- Dr. Ryan Stansifer *Client/Advisor*
- Dr. Philip Chan *CSE4201 Instructor*

Motivation

- Tedious debugging tasks
  - Time wasted on code tracing
  - Need to keep track of data movement
- Unintuitive UI design of modern debuggers
  - GDB
  - WinDBG
  - Radare

Goal

- Automatic data visualization
  - Code tracing becomes unnecessary
  - Data movement is animated between steps
- Simple, yet effective GUI
  - No need to memorize commands
  - Accomplish complex tasks more quickly
  - Look pretty :)

Key Features

- Interactive GUI
  - Automatically generate data structure diagrams
  - Animate data movement between steps
- Dynamic code analysis
  - Step line-by-line through source code
- User intervention
  - Manually override incorrect data structure diagrams
  - Choose how certain structures are represented

Algorithms and Tools

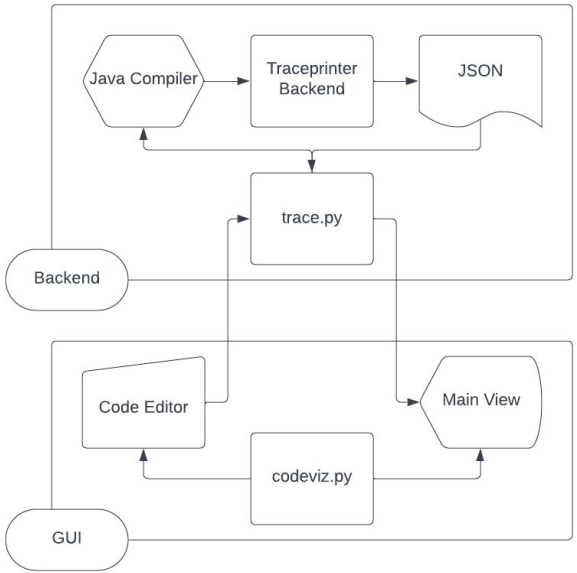
- PyQt
  - Widget-based GUI framework
  - Written entirely in Python 3
  - Cross-platform compatibility
  - Custom widgets for each data structure
- Traceprinter
  - Backend execution tracing borrowed from *Java Visualizer*
  - Perfectly matches our needs after some slight modification

Technical Challenges

- Integrating backend and frontend may prove difficult
- Inexperience with GUI development
- Client requested compatibility with modern versions of Java
    - Deprecated libraries
    - Unfamiliarity with Traceprinter's source code

System Architecture Diagram

- trace.py
  - Interface between GUI and backend
  - Compiled Java code runs in Traceprinter environment
    - Produces JSON output
    - Returned to trace.py
- codeviz.py
  - Contains all GUI functionality
  - Sends data to trace.py for processing



Evaluation

- Speed

  How long does it take to fully generate the visual elements after submitting code?

- Reliability

  How often does the system correctly identify data structure types?

Progress Summary

Module/feature	Completion %	To do
Traceprinter backend	90%	Integrate with frontend
GUI	0%	Everything
Custom data structures	10%	Write the rest of the data structures listed in the requirements document

Milestone 4

- Set up main window in PyQt
- Implement code editor
- Write custom List and Map implementations
- Modify Traceprinter to add multiple files to classpath

Milestone 5

- Implement data structure diagrams
- Conduct evaluation and analyze results
- Create poster and ebook page for Senior Design Showcase

Milestone 6

- Implement data structure diagrams
- Test/demo of the entire system
- Conduct evaluation and analyze results
- Create user/developer manual
- Create demo video

Task Matrix for Milestone 4

Task	Curtice	Josh	Catherine
1. PyQt main window	50%	50%	0%
2. Implement code editor	50%	50%	0%
3. Custom List/Map implementations	0%	0%	100%
4. Modify Traceprinter compile-time options	100%	0%	0%

Thank You