Bachelor's Thesis

JOHANNES KEPLER
UNIVERSITY LINZ

Visual Studio Code Plugin for Debugging Programs on GraalVM

Student: Lukas Freiseisen
SKZ/Matr.Nr.: 521 / k01168215
Email: lukas.freiseisen@gmail.com

Advisor: DI Jacob Kreindl, Bsc.

Start Date: Feb. 6, 2019

DI Jacob Kreindl, Bsc.
Institute for System Software

P +43 732 2468 4370
F +43 732 2468 4345
jacob.kreindl@jku.at

Office:
Karin Gusenbauer
Ext. 4342
karin.gusenbauer@jku.at

Truffle [1,2] is a framework for implementing interpreters for programming languages. It is the key component of the GraalVM [3] project which combines several such Truffle language implementations into a single multi-language runtime. The framework also provides an API for efficient instrumentation and debugging of programs executed by a Truffle language implementation [4]. Multiple debuggers can already use this API as a back-end, e.g., the debugger of the NetBeans IDE [5] and GraalVM's integrated support for the Chrome Developer Tools Protocol [6].

Visual Studio Code [7] is a code editor with support for multiple programming languages and advanced IDE-like features that can be further extended using plug-ins. One such feature is an integrated debugger front-end. Several popular plug-ins integrate this plug-in with debugger back-ends provided by existing language run-times.

The goal of this thesis is to develop a plugin for Visual Studio Code that uses Truffle's debugging framework as a back-end for the editor's integrated debugger. As a result, users should be able to use Visual Studio code to debug programs executed on GraalVM.

The concrete goals of the thesis are as follows:

- Develop a plug-in for Visual Studio Code that connects its integrated debugger front-end to a back-end that uses GraalVM's debugging framework to control and inspect programs executed by GraalVM.
- Support basic debugger features:
  - Inspecting the scope and symbols valid at the position at which the program is suspended
  - Inspecting the current call-stack
  - Setting, enabling (and halting the program at), disabling and removing line-based breakpoints.

JOHANNES KEPLER
UNIVERSITY LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at
DVR 0093696

    - Stepping (stepping into and out of function calls, and single-stepping over statements of the current function)
    - Evaluate user-supplied expressions in the context of a halted function
    - View loaded source files
- Provide a user-friendly way of launching the program to debug
    - A user should be able to run a program with defined arguments on GraalVM and launch the debugger all from within Visual Studio Code.
    - A user should be able to launch a debugger session in which they are able to interactively select pieces of code from within an existing file to evaluate and inspect the result. This interactive session may be restricted to a single programming language per session.
- Provide an automated framework to test all implemented features.

Optional goals for this project are as follows:

- Support column-based and/or function-based breakpoints.
- Support conditional breakpoints.
- Support multiple languages simultaneously for expression evaluation in an interactive session.

Explicit non-goals of this thesis are:

- Searching for and/or fixing bugs in GraalVM.
    - However, any bugs that are found should be reported.
- Extending GraalVM's instrumentation or debugging framework. Should any of the listed goals not be reachable due to limitations or deficiencies in this framework, the scope of the project may be adapted after consultation with the advisor. However, a description of the limitation and how it prevents a certain feature to be implemented may be part of the written thesis.

The progress of the thesis should be discussed with the adviser on at least a bi-weekly basis. The final version of the written thesis should be submitted before September 30, 2019.

[1] T. Würthinger et al., “One VM to Rule Them All,” in Proceedings of the 2013 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software, New York, NY, USA, 2013, pp. 187–204.

[2] https://github.com/oracle/graal/tree/master/truffle

[3] https://www.graalvm.org/

[4] M. Van De Vanter, C. Seaton, M. Haupt, C. Humer, and T. Würthinger, “Fast, Flexible, Polyglot Instrumentation Support for Debuggers and other Tools,” The Art, Science, and Engineering of Programming, vol. 2, no. 3, pp. 14:1-14:30, Mar. 2018.

[5] https://netbeans.org/

[6] https://www.graalvm.org/docs/reference-manual/tools/#debugger

[7] https://code.visualstudio.com/