JavaScript Profiler

Roy Adams and David Belinger

October 18, 2013

1 Overview

Our profiler takes as input a block of JavaScript code as a string, adds instrumentation code to all user defined functions, and then runs the code to collect profile data. The profiler may be called either as an imported library or via a simple HTML interface that allows the user to paste in JavaScript code he or she wants to profile. During a run of the input script, the profiler collects timing, call-path, and call frequency information which can then be accessed as a formatted output.

2 Design

The profiler is defined completely in profiling.ts (which typescript then compiles to profiling.js) and consists of three main classes: ProfileFromSource, Profiler, and Profile. The top level class is ProfileFromSource which serves as an interface to the user. It has only three functions, a constructor which modifies the input code, startUp which runs the modified code and collects profile information, and getReport which compiles the profile information and returns a formatted string. Underneath this is the Profiler class which does most of the leg-work in our profiler. The Profiler class stores and maintains all global profile objects such as the call stack, call paths, and a list of defined functions. Finally, the Profile class maintains information about a specific function such as average time spent in the function and number of invocations.

3 Interface

To use the profiler as a library

4 Discussion