Table of Contents

[1 Cheat Sheet 2](#_Toc14005117)

[1.1 outdated library 2](#_Toc14005118)

[2 Take Advantage of the Built-in Profiler 2](#_Toc14005119)

[2.1 Running your application with profiling enabled 2](#_Toc14005120)

[2.2 Processing your tick file 2](#_Toc14005121)

[2.3 Reading the tick file processing result 2](#_Toc14005122)

# Cheat Sheet

## outdated library

npm outdated

# Take Advantage of the Built-in Profiler

Performance is very important to any system. Profiling your code will allow you to understand where time is spent in your code, so you can focus on the right pieces of code that can most benefit from performance optimization.

## Running your application with profiling enabled

It’s very easy to start using profiler; simply add the --prof option to your node command:

npm --prof start

Execute your code. When finished, the profiler will create a “tick file”; a big text file in your current working directory named isolate-0x???????-v8.log. We need to process this file.

## Processing your tick file

To process your tick file, simply run the following command:

node --prof-process isolate-0x???????-v8.log

## Reading the tick file processing result

The processed result will contain several parts, including:

[Shared libraries]

[JavaScript]

[C++]

[Summary]

[C++ entry points]

[Bottom up (heavy) profile]

[Summary] part it is short and gives an immediate overview of where to dig further