**Memory Leak Workshop:**

You have two servers, each one of them has a memory leak.

In order to see and reproduce the leak – you will have to send multiple requests to the each server – only then the leak will start to appear.

Server 1 appears in ex1.

Server 2 appears in

<https://github.com/tamartwe/server_with_leak>

Note:

I would like you to profile both servers, because on each server the leak is different and you will see something different in the traces.

For server1 get request should be sufficient to reproduce the leak. Do several get requests.

For server2 you need to make about 100 post requests to reproduce the leak.

For each server, do the following operations.

* First – add a function to print process.memoryUsage and see it grows. Try the function will run randomly and not on constant time

<https://nodejs.org/api/process.html#process_process_memoryusage>

* Work with memwatch-next, listen to the leak event and see the print. Look at the class presentation to see how to work with it.

<https://www.npmjs.com/package/memwatch-next>

* Take heapdump snapshot, open it in chrome development tools, try to feagure out where the problem.

<https://www.npmjs.com/package/heapdump>

* Also – run the server with –inspect to open inspector and take a snapshot from there.

How to run the node –inspect flag :

Run the program with node –inspect <main\_script.ja>

Concrete Example :

When running with –inspect flag – you will get:

* + Debugger listening on ws://127.0.0.1:9229/da161d57-bee0-4a2b-a71e-42cb39711875

Use this prefix url in browser to connect:

* + chrome-devtools://devtools/bundled/inspector.html?experiments=true&v8only=true&ws=127.0.0.1:9229/da161d57-bee0-4a2b-a71e-42cb39711875

Open chrome devtools :

Open [**Google Chrome**](https://www.google.com/intl/en/chrome/browser/) and press F12 to open the developer toolbar.

Go to the Profiles tab, right-click in the tab pane and select Load profile....

Select the dump file and click Open. You can now inspect the heap snapshot at your leisure.

Note that Chrome will refuse to load the file unless it has the .heapsnapshot extension.

* Fix the leak – and to verify your solution do all the steps – see memwatch behaving better , see trace with the fix, compare the trace to the trace before the fix.