

Node sin consola

tatistical profiling result from sinConsola.log, (4183 ticks, 0 unaccounted, 0 excluded).

[Shared libraries]:

ticks	total	nonlib	name
2808	67.1%		C:\WINDOWS\SYSTEM32\ntdll.dll
1366	32.7%		C:\Program Files\nodejs\node.exe
2	0.0%		C:\WINDOWS\System32\KERNELBASE.dll

[JavaScript]:

ticks	total	nonlib	name
6	0.1%	85.7%	LazyCompile: *resolve path.js:153:10
1	0.0%	14.3%	LazyCompile: *isFileType fs.js:199:20

[C++]:

ticks	total	nonlib	name
-------	-------	--------	------

[Summary]:

ticks	total	nonlib	name
7	0.2%	100.0%	JavaScript
0	0.0%	0.0%	C++
10	0.2%	142.9%	GC
4176	99.8%		Shared libraries

[C++ entry points]:

ticks	cpp	total	name
-------	-----	-------	------

[Bottom up (heavy) profile]:

Note: percentage shows a share of a particular caller in the total

amount of its parent calls.

Callers occupying less than 1.0% are not shown.

ticks parent name

2808	67.1%	C:\WINDOWS\SYSTEM32\ntdll.dll
402	14.3%	LazyCompile: *readFileSync fs.js:391:22
402	100.0%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
312	77.6%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
312	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
312	100.0%	LazyCompile: ~require internal/modules/cjs/helpers.js:91:31
90	22.4%	LazyCompile: ~Module.load internal/modules/cjs/loader.js:925:33
90	100.0%	LazyCompile: ~Module._load internal/modules/cjs/loader.js:724:24
90	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
219	7.8%	LazyCompile: *stat internal/modules/cjs/loader.js:145:14
143	65.3%	LazyCompile: ~tryFile internal/modules/cjs/loader.js:351:17
139	97.2%	LazyCompile: ~tryExtensions internal/modules/cjs/loader.js:367:23
112	80.6%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
112	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
27	19.4%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
22	81.5%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
5	18.5%	LazyCompile: *Module._findPath internal/modules/cjs/loader.js:461:28
4	2.8%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
4	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
4	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
49	22.4%	LazyCompile: *Module._findPath internal/modules/cjs/loader.js:461:28
49	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
49	100.0%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
49	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36

27	12.3%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
27	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
25	92.6%	LazyCompile: ~Module._load internal/modules/cjs/loader.js:724:24
25	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
2	7.4%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
2	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
1366	32.7%	C:\Program Files\nodejs\node.exe
1309	95.8%	C:\Program Files\nodejs\node.exe
815	62.3%	LazyCompile: ~openSync fs.js:489:18
815	100.0%	LazyCompile: ~readFileSync fs.js:391:22
813	99.8%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
813	100.0%	LazyCompile: ~Module.load internal/modules/cjs/loader.js:925:33
253	19.3%	LazyCompile: ~realpathSync fs.js:1718:22
216	85.4%	LazyCompile: ~toRealPath internal/modules/cjs/loader.js:360:20
208	96.3%	LazyCompile: ~tryFile internal/modules/cjs/loader.js:351:17
190	91.3%	LazyCompile: ~tryExtensions internal/modules/cjs/loader.js:367:23
18	8.7%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
8	3.7%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
8	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
37	14.6%	LazyCompile: *Module._findPath internal/modules/cjs/loader.js:461:28
37	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
37	100.0%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
97	7.4%	LazyCompile: ~stat internal/modules/cjs/loader.js:145:14
71	73.2%	LazyCompile: ~tryFile internal/modules/cjs/loader.js:351:17
66	93.0%	LazyCompile: ~tryExtensions internal/modules/cjs/loader.js:367:23
52	78.8%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
14	21.2%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20

5	7.0%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
5	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
26	26.8%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
25	96.2%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
25	100.0%	LazyCompile: ~Module._load internal/modules/cjs/loader.js:724:24
1	3.8%	LazyCompile: ~resolveMainPath internal/modules/run_main.js:12:25
1	100.0%	LazyCompile: ~executeUserEntryPoint internal/modules/run_main.js:69:31
35	2.7%	LazyCompile: ~wrapSafe internal/modules/cjs/loader.js:973:18
35	100.0%	LazyCompile: ~Module._compile internal/modules/cjs/loader.js:1026:37
35	100.0%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
35	100.0%	LazyCompile: ~Module.load internal/modules/cjs/loader.js:925:33
32	2.4%	LazyCompile: ~read internal/modules/package_json_reader.js:16:14
30	93.8%	LazyCompile: ~readPackage internal/modules/cjs/loader.js:257:21
27	90.0%	LazyCompile: ~resolveExports internal/modules/cjs/loader.js:439:24
27	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
3	10.0%	LazyCompile: ~readPackageScope internal/modules/cjs/loader.js:288:26
3	100.0%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
2	6.3%	LazyCompile: *readPackage internal/modules/cjs/loader.js:257:21
2	100.0%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
2	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28

Node con consola

Statistical profiling result from conConsola.log, (4433 ticks, 0 unaccounted, 0 excluded).

[Shared libraries]:

ticks	total	nonlib	name
-------	-------	--------	------

3079	69.5%		C:\WINDOWS\SYSTEM32\ntdll.dll
------	-------	--	-------------------------------

1349	30.4%		C:\Program Files\nodejs\node.exe
------	-------	--	----------------------------------

[JavaScript]:

ticks	total	nonlib	name
-------	-------	--------	------

3	0.1%	60.0%	LazyCompile: *resolve path.js:153:10
2	0.0%	40.0%	LazyCompile: *readPackageScope internal/modules/cjs/loader.js:288:26

[C++]:

ticks	total	nonlib	name
-------	-------	--------	------

[Summary]:

ticks	total	nonlib	name
-------	-------	--------	------

5	0.1%	100.0%	JavaScript
0	0.0%	0.0%	C++
13	0.3%	260.0%	GC
4428	99.9%		Shared libraries

[C++ entry points]:

ticks	cpp	total	name
-------	-----	-------	------

[Bottom up (heavy) profile]:

Note: percentage shows a share of a particular caller in the total amount of its parent calls.

Callers occupying less than 1.0% are not shown.

ticks	parent	name
-------	--------	------

3079	69.5%	C:\WINDOWS\SYSTEM32\ntdll.dll
399	13.0%	LazyCompile: *readFileSync fs.js:391:22
398	99.7%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
304	76.4%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
304	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36

304	100.0%	LazyCompile: ~require internal/modules/cjs/helpers.js:91:31
94	23.6%	LazyCompile: ~Module.load internal/modules/cjs/loader.js:925:33
94	100.0%	LazyCompile: ~Module._load internal/modules/cjs/loader.js:724:24
94	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
203	6.6%	LazyCompile: *stat internal/modules/cjs/loader.js:145:14
110	54.2%	LazyCompile: ~tryFile internal/modules/cjs/loader.js:351:17
106	96.4%	LazyCompile: ~tryExtensions internal/modules/cjs/loader.js:367:23
91	85.8%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
91	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
15	14.2%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
10	66.7%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
5	33.3%	LazyCompile: *Module._findPath internal/modules/cjs/loader.js:461:28
3	2.7%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
3	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
3	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
58	28.6%	LazyCompile: *Module._findPath internal/modules/cjs/loader.js:461:28
58	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
58	100.0%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
58	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
35	17.2%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
35	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
32	91.4%	LazyCompile: ~Module._load internal/modules/cjs/loader.js:724:24
32	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
3	8.6%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
3	100.0%	LazyCompile: ~Module.require internal/modules/cjs/loader.js:953:36
1349	30.4%	C:\Program Files\nodejs\node.exe
1272	94.3%	C:\Program Files\nodejs\node.exe

769	60.5%	LazyCompile: ~openSync fs.js:489:18
769	100.0%	LazyCompile: ~readFileSync fs.js:391:22
767	99.7%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
767	100.0%	LazyCompile: ~Module.load internal/modules/cjs/loader.js:925:33
252	19.8%	LazyCompile: ~realpathSync fs.js:1718:22
212	84.1%	LazyCompile: ~toRealPath internal/modules/cjs/loader.js:360:20
201	94.8%	LazyCompile: ~tryFile internal/modules/cjs/loader.js:351:17
181	90.0%	LazyCompile: ~tryExtensions internal/modules/cjs/loader.js:367:23
20	10.0%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
11	5.2%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
10	90.9%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
1	9.1%	LazyCompile: ~resolveMainPath internal/modules/run_main.js:12:25
40	15.9%	LazyCompile: *Module._findPath internal/modules/cjs/loader.js:461:28
40	100.0%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
40	100.0%	LazyCompile: *Module._load internal/modules/cjs/loader.js:724:24
105	8.3%	LazyCompile: ~stat internal/modules/cjs/loader.js:145:14
70	66.7%	LazyCompile: ~tryFile internal/modules/cjs/loader.js:351:17
60	85.7%	LazyCompile: ~tryExtensions internal/modules/cjs/loader.js:367:23
56	93.3%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
4	6.7%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
10	14.3%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
10	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
35	33.3%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
34	97.1%	LazyCompile: ~Module._resolveFilename internal/modules/cjs/loader.js:804:35
34	100.0%	LazyCompile: ~Module._load internal/modules/cjs/loader.js:724:24
1	2.9%	LazyCompile: ~resolveMainPath internal/modules/run_main.js:12:25
1	100.0%	LazyCompile: ~executeUserEntryPoint internal/modules/run_main.js:69:31
49	3.9%	LazyCompile: ~wrapSafe internal/modules/cjs/loader.js:973:18

49	100.0%	LazyCompile: ~Module._compile internal/modules/cjs/loader.js:1026:37
49	100.0%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
49	100.0%	LazyCompile: ~Module.load internal/modules/cjs/loader.js:925:33
35	2.8%	LazyCompile: ~read internal/modules/package_json_reader.js:16:14
32	91.4%	LazyCompile: ~readPackage internal/modules/cjs/loader.js:257:21
28	87.5%	LazyCompile: ~resolveExports internal/modules/cjs/loader.js:439:24
28	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28
4	12.5%	LazyCompile: ~readPackageScope internal/modules/cjs/loader.js:288:26
4	100.0%	LazyCompile: ~Module._extensions..js internal/modules/cjs/loader.js:1081:37
3	8.6%	LazyCompile: *readPackage internal/modules/cjs/loader.js:257:21
3	100.0%	LazyCompile: ~tryPackage internal/modules/cjs/loader.js:305:20
3	100.0%	LazyCompile: ~Module._findPath internal/modules/cjs/loader.js:461:28

Artillery salida sin console.log

Started phase 0, duration: 1s @ 11:38:05(-0300) 2021-08-22

Report @ 11:38:15(-0300) 2021-08-22

Elapsed time: 10 seconds

Scenarios launched: 20

Scenarios completed: 0

Requests completed: 195

Mean response/sec: 20

Response time (msec):

min: 18

max: 1072

median: 449

p95: 835.5

p99: 1016.6

Codes:



200: 195

Report @ 11:38:25(-0300) 2021-08-22

Elapsed time: 20 seconds

Scenarios launched: 0

Scenarios completed: 0

Requests completed: 202

Mean response/sec: 20.93

Response time (msec):

min: 214

max: 1650

median: 443.5

p95: 976

p99: 1257.8

Codes:

200: 202

Report @ 11:38:35(-0300) 2021-08-22

Elapsed time: 30 seconds

Scenarios launched: 0

Scenarios completed: 0

Requests completed: 196

Mean response/sec: 19.52

Response time (msec):

min: 261

max: 1191

median: 487

p95: 1037.3

p99: 1148.8

Codes:

200: 196

Report @ 11:38:45(-0300) 2021-08-22

Elapsed time: 40 seconds

Scenarios launched: 0

Scenarios completed: 0

Requests completed: 200

Mean response/sec: 20.22

Response time (msec):

min: 296

max: 1269

median: 471

p95: 875

p99: 1121

Codes:

200: 200

Report @ 11:38:55(-0300) 2021-08-22

Elapsed time: 50 seconds

Scenarios launched: 0

Scenarios completed: 2

Requests completed: 199

Mean response/sec: 19.51

Response time (msec):

min: 265

max: 1357

median: 447

p95: 1052.2

p99: 1347.1

Codes:

200: 199

Report @ 11:38:56(-0300) 2021-08-22

Elapsed time: 51 seconds

Scenarios launched: 0

Scenarios completed: 18

Requests completed: 8

Mean response/sec: 4.39

Response time (msec):

min: 292

max: 686

median: 382

p95: 686

p99: 686

Codes:

200: 8

All virtual users finished

Summary report @ 11:38:56(-0300) 2021-08-22

Scenarios launched: 20

Scenarios completed: 20

Requests completed: 1000

Mean response/sec: 19.54

Response time (msec):

min: 18

max: 1650

median: 455

p95: 951.5

p99: 1223.5

Scenario counts:

0: 20 (100%)

Codes:

200: 1000

Artillery salida con console.log

Started phase 0, duration: 1s @ 11:36:17(-0300) 2021-08-22

Report @ 11:36:27(-0300) 2021-08-22

Elapsed time: 10 seconds

Scenarios launched: 20

Scenarios completed: 0

Requests completed: 205

Mean response/sec: 21.95

Response time (msec):

min: 7

max: 983

median: 382

p95: 860.3

p99: 981.5

Codes:

200: 205

Report @ 11:36:37(-0300) 2021-08-22

Elapsed time: 20 seconds

Scenarios launched: 0

Scenarios completed: 0

Requests completed: 223

Mean response/sec: 21.85

Response time (msec):

min: 324

max: 1483

median: 379

p95: 852.5

p99: 1038.9

Codes:

200: 223

Autocannon sin consola:

Running 20s test @ http://localhost:8080/info  
100 connections

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	1774 ms	1936 ms	3416 ms	3609 ms	2032.38 ms	364.45 ms	3794 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	0	0	52	58	46.85	15.3	6
Bytes/Sec	0 B	0 B	136 kB	152 kB	123 kB	40 kB	15.7 kB

Req/Bytes counts sampled once per second.

Autocannon con consola

Running 20s test @ http://localhost:8080/info?console=true  
100 connections

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	1954 ms	2130 ms	3562 ms	3753 ms	2387.99 ms	476.9 ms	4134 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	0	0	44	52	34.65	18.42	10
Bytes/Sec	0 B	0 B	115 kB	136 kB	90.7 kB	48.2 kB	26.2 kB

Req/Bytes counts sampled once per second.

793 requests in 20.19s, 1.81 MB read

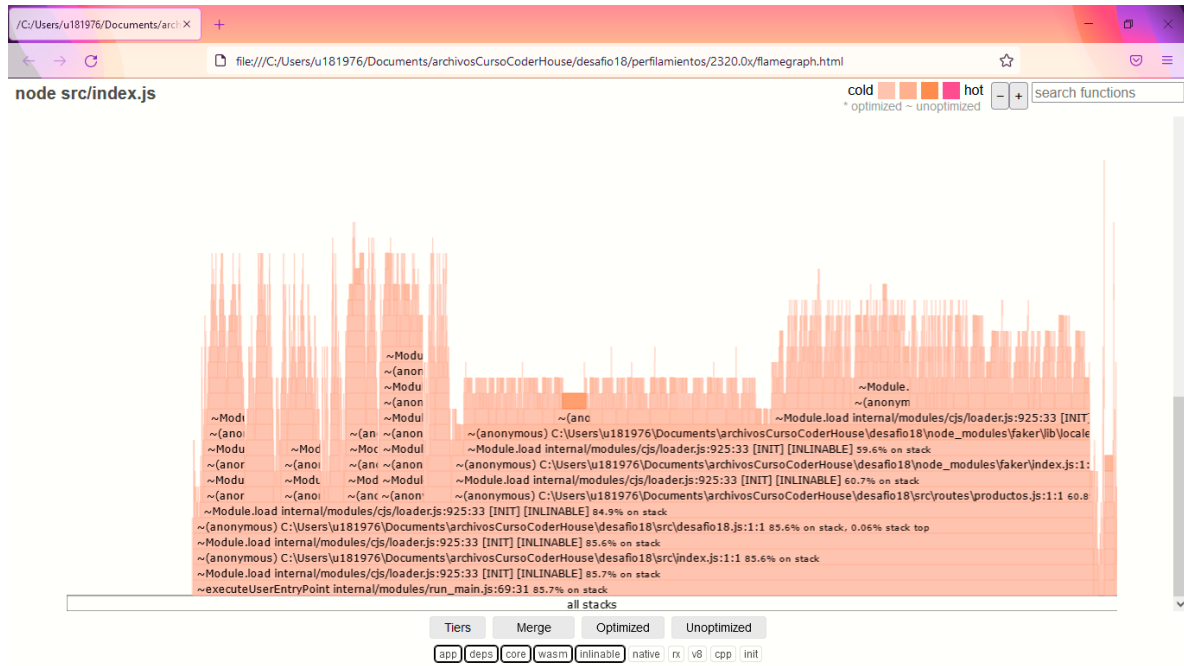
## Inspect Autocannon sin consola

Profiles	Self Time	Total Time	Function
CPU PROFILES	38348.4 ms 83.09 %	38348.4 ms 83.09 %	(program)
Profile 1	390.5 ms 0.85 %	416.8 ms 0.90 %	next
	300.7 ms 0.65 %	300.7 ms 0.65 %	writeUtf8String
	226.1 ms 0.49 %	1049.6 ms 2.27 %	SourceNode_walk
	204.8 ms 0.44 %	204.8 ms 0.44 %	(garbage collector)
	198.5 ms 0.43 %	271.4 ms 0.59 %	SourceNode_add
	195.2 ms 0.42 %	775.6 ms 1.60 %	parse
	146.4 ms 0.32 %	146.4 ms 0.32 %	stat
	136.4 ms 0.30 %	658.3 ms 1.43 %	wrap
	132.1 ms 0.29 %	132.1 ms 0.29 %	quotedString
	130.8 ms 0.28 %	130.8 ms 0.28 %	writeBuffer
	125.9 ms 0.27 %	453.3 ms 0.98 %	createFunctionContext
	116.0 ms 0.25 %	116.0 ms 0.25 %	writev
	96.4 ms 0.21 %	1787.4 ms 3.87 %	compile
	95.4 ms 0.21 %	451.3 ms 0.98 %	replaceStack
	74.9 ms 0.16 %	3248.8 ms 7.04 %	render
	56.6 ms 0.12 %	65.1 ms 0.14 %	nextTick
	55.5 ms 0.12 %	229.9 ms 0.50 %	accept
	49.5 ms 0.11 %	113.7 ms 0.25 %	anonymous
	47.6 ms 0.10 %	483.4 ms 1.05 %	accept
	47.6 ms 0.10 %	4434.5 ms 9.61 %	next
	46.7 ms 0.10 %	162.9 ms 0.35 %	deserializeObject

## Inspect Autocannon con consola

Profiles	Self Time	Total Time	Function
CPU PROFILES	39399.8 ms 77.58 %	39399.8 ms 77.58 %	(program)
Profile 1	2064.3 ms 4.06 %	2064.3 ms 4.06 %	writeUtf8String
	2064.3 ms 4.06 %	2064.3 ms 4.06 %	handleWriteReq
	2064.3 ms 4.06 %	2064.3 ms 4.06 %	writeGeneric
Profile 2	2064.3 ms 4.06 %	2064.3 ms 4.06 %	Socket_writeGeneric
	882.2 ms 1.74 %	3012.7 ms 5.93 %	consoleCall
	882.2 ms 1.74 %	3012.7 ms 5.93 %	(anonymous)
	882.2 ms 1.74 %	3012.7 ms 5.93 %	handle
	882.2 ms 1.74 %	3012.7 ms 5.93 %	next
	269.6 ms 0.53 %	410.6 ms 0.81 %	next
	251.1 ms 0.49 %	1188.2 ms 2.34 %	SourceNode_walk
	236.7 ms 0.47 %	838.4 ms 1.65 %	parse
	235.8 ms 0.46 %	235.8 ms 0.46 %	(garbage collector)
	212.1 ms 0.42 %	290.2 ms 0.57 %	SourceNode_add
	157.3 ms 0.31 %	157.3 ms 0.31 %	stat
	145.4 ms 0.29 %	145.4 ms 0.29 %	quotedString
	135.4 ms 0.27 %	135.4 ms 0.27 %	writeBuffer
	131.7 ms 0.26 %	691.1 ms 1.36 %	wrap
	120.0 ms 0.24 %	481.4 ms 0.95 %	createFunctionContext
	118.8 ms 0.23 %	118.8 ms 0.23 %	writev
	97.1 ms 0.19 %	1881.5 ms 3.70 %	compile
	89.4 ms 0.18 %	451.7 ms 0.89 %	replaceStack

## Flamegraph sin consola



## Flamegraph con consola



## Comandos y conclusión

/ Info sin consoleLog

```
curl -X GET "http://localhost:8080/info"
```

// Info con consoleLog

```
curl -X GET "http://localhost:8080/info?console=true"
```

//Ejecutar servidor en modo prof

```
node --prof src/index.js
```

//Test de Carga con Artillery

```
artillery quick --count 20 -n 50 "http://localhost:8080/info" > artillery_sinConsole.txt
```

```
artillery quick --count 20 -n 50 "http://localhost:8080/info?console=true" >  
artillery_ConConsole.txt
```

//Parsear resultados con --Prof-Process

```
node --prof-process sinConsola.log > v8SinConsola.txt
```

```
node --prof-process conConsola.log > v8SConConsola.txt
```

----

```
node --inspect src/index.js
```

```
autocannon -c 100 -d 20 "http://localhost:8080/info"
```

```
autocannon -c 100 -d 20 "http://localhost:8080/info?console=true"
```

```
0x -P 'autocannon -c 100 -d 20 "http://localhost:8080/info"' src/index.js
```



```
0x -P 'autocannon -c 100 -d 20 "http://localhost:8080/info?console=true"' src/index.js
```

-----  
-----

## Conclusión

Se verifica en todos los test que el console.log hace más lenta la ejecución.