

Comparativa

Análisis de rendimiento

Silvia Colmenares

Prueba Artillery Con Console log

Phase started: unnamed (index: 0, duration: 1s) 20:45:31(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 20:45:32(-0300)

Metrics for period to: 20:45:40(-0300) (width: 1.852s)

http.codes.200:	1000
http.request_rate:	548/sec
http.requests:	1000
http.response_time:	
min:	5
max:	123
median:	46.1
p95:	77.5
p99:	87.4
http.responses:	1000
vusers.completed:	50
vusers.created:	50
vusers.created_by_name.0:	50
vusers.failed:	0
vusers.session_length:	
min:	374.5
max:	1161.3
median:	1022.7
p95:	1130.2
p99:	1153.1

Prueba Artillery Sin Console log

Phase started: unnamed (index: 0, duration: 1s) 20:46:02(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 20:46:03(-0300)

Metrics for period to: 20:46:10(-0300) (width: 1.564s)

http.codes.200:	1000
http.request_rate:	649/sec
http.requests:	1000
http.response_time:	
min:	4
max:	76
median:	32.1
p95:	48.9
p99:	54.1
http.responses:	1000
vusers.completed:	50
vusers.created:	50
vusers.created_by_name.0:	50
vusers.failed:	0
vusers.session_length:	
min:	218.5
max:	811.6
median:	658.6
p95:	788.5
p99:	788.5

Profile Con Console log

Statistical profiling result from .\insolateConConsole.log, (4547 ticks, 0 unaccounted, 0 excluded).

[Shared libraries]:

ticks	total	nonlib	name
3993	87.8%		C:\WINDOWS\SYSTEM32\ntdll.dll
538	11.8%		C:\Program Files\nodejs\node.exe
3	0.1%		C:\WINDOWS\System32\KERNELBASE.dll
1	0.0%		C:\WINDOWS\System32\KERNEL32.DLL

[JavaScript]:

ticks	total	nonlib	name
3	0.1%	25.0%	LazyCompile: *resolve node:path:158:10
1	0.0%	8.3%	LazyCompile: *next C:\Users\SCON\Desktop\backen\node_modules\express\lib\router\index.js:177:16
1	0.0%	8.3%	LazyCompile: *emit node:events:470:44
1	0.0%	8.3%	LazyCompile: *dirname node:path:653:10
1	0.0%	8.3%	LazyCompile: *compileFunction node:vm:308:25
1	0.0%	8.3%	LazyCompile: *Module._load node:internal/modules/cjs/loader:757:24
1	0.0%	8.3%	LazyCompile: *Module._compile node:internal/modules/cjs/loader:1057:37
1	0.0%	8.3%	Function: ^realpathSync node:fs:2410:22
1	0.0%	8.3%	Function: ^onwrite C:\Users\SCON\Desktop\backen\node_modules\winston\node_modules\readable-stream\lib_stream_writable.js:4
1	0.0%	8.3%	Function: ^initialize C:\Users\SCON\Desktop\backen\node_modules\passport\lib\middleware\initialize.js:51:29

Profile Sin Console log

Statistical profiling result from .\insolateSinConsole.log, (9354 ticks, 0 unaccounted, 0 excluded).

[Shared libraries]:

ticks	total	nonlib	name
8782	93.9%		C:\WINDOWS\SYSTEM32\ntdll.dll
546	5.8%		C:\Program Files\nodejs\node.exe
3	0.0%		C:\WINDOWS\System32\KERNELBASE.dll

[JavaScript]:

ticks	total	nonlib	name
7	0.1%	30.4%	LazyCompile: *resolve node:path:158:10
2	0.0%	8.7%	LazyCompile: *processTicksAndRejections node:internal/process/task_queues:68:35
1	0.0%	4.3%	RegExp: @@@
1	0.0%	4.3%	RegExp: ; *([!#\$%&'*.^_`~0-9A-Za-z-]+) *= *("(?:[\u000b\u0020\u0021\u0023-\u005b\u005d-\u007e\u0080-\u00ff] \\"
1	0.0%	4.3%	LazyCompile: *slowCases node:internal/util:165:19
1	0.0%	4.3%	LazyCompile: *readPackageScope node:internal/modules/cjs/loader:321:26
1	0.0%	4.3%	LazyCompile: *dirname node:path:653:10
1	0.0%	4.3%	LazyCompile: *Module._nodeModulePaths node:internal/modules/cjs/loader:583:37
1	0.0%	4.3%	LazyCompile: *Module._load node:internal/modules/cjs/loader:757:24
1	0.0%	4.3%	Function: ^wrapmethods C:\Users\SCON\Desktop\backen\node_modules\express-session\index.js:394:25
1	0.0%	4.3%	Function: ^randomBytesSync C:\Users\SCON\Desktop\backen\node_modules\random-bytes\index.js:72:25
1	0.0%	4.3%	Function: ^isBuffer node:buffer:509:36
1	0.0%	4.3%	Function: ^compileForInternalLoader node:internal/bootstrap/loaders:299:27
1	0.0%	4.3%	Function: ^Module._load node:internal/modules/cjs/loader:757:24
1	0.0%	4.3%	Function: ^<anonymous> node:internal/validators:168:39
1	0.0%	4.3%	Function: ^<anonymous> node:internal/fs/utils:357:35

Autocannon Con Console log

```
SCON@LAPTOP-FAKL3LAP MINGW64 ~/Desktop/backen (Tarea12nginx)
```

```
$ npm run autocannon
```

```
> tarea-3@1.0.0 autocannon
```

```
> autocannon -c 100 -d 20 http://localhost:8080/api/process/info
```

```
Running 20s test @ http://localhost:8080/api/process/info
```

```
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	79 ms	92 ms	138 ms	143 ms	96.79 ms	16.36 ms	203 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	693	693	1074	1125	1025.95	116.8	693
Bytes/Sec	569 kB	569 kB	883 kB	926 kB	844 kB	96.2 kB	569 kB

```
Req/Bytes counts sampled once per second.
```

```
# of samples: 20
```

```
21k requests in 20.04s, 16.9 MB read
```

```
SCON@LAPTOP-FAKL3LAP MINGW64 ~/Desktop/backen (Tarea12nginx)
```

```
$ Con console
```

Autocannon Sin Console log

```
$ npm run autocannon
```

```
> tarea-3@1.0.0 autocannon
```

```
> autocannon -c 100 -d 20 http://localhost:8080/api/process/info
```

```
Running 20s test @ http://localhost:8080/api/process/info
```

```
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	67 ms	79 ms	123 ms	131 ms	83.47 ms	16.08 ms	174 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	795	795	1143	1380	1189.3	163.03	795
Bytes/Sec	653 kB	653 kB	941 kB	1.14 MB	978 kB	134 kB	653 kB

```
Req/Bytes counts sampled once per second.
```

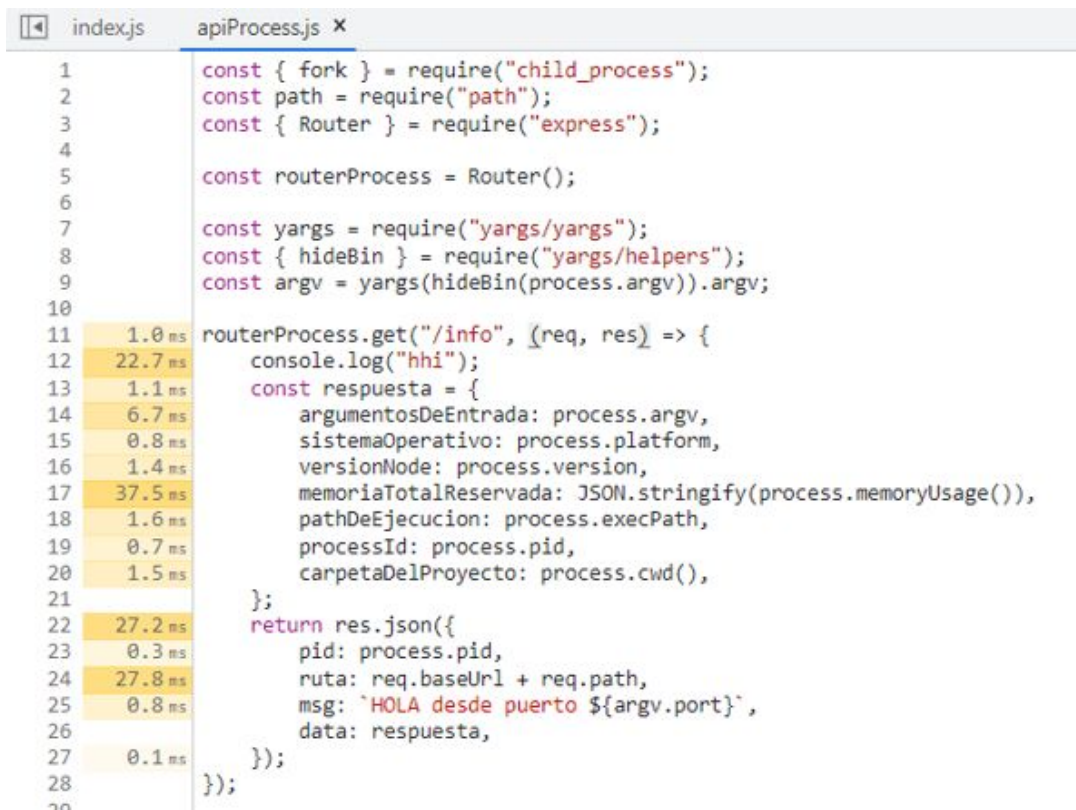
```
# of samples: 20
```

```
24k requests in 20.05s, 19.6 MB read
```

```
SCON@LAPTOP-FAKL3LAP MINGW64 ~/Desktop/backen (Tarea12nginx)
```

```
$ SIn console
```

Inspect Con Console log



```
index.js  apiProcess.js x
1      const { fork } = require("child_process");
2      const path = require("path");
3      const { Router } = require("express");
4
5      const routerProcess = Router();
6
7      const yargs = require("yargs/yargs");
8      const { hideBin } = require("yargs/helpers");
9      const argv = yargs(hideBin(process.argv)).argv;
10
11      1.0 ms routerProcess.get("/info", (req, res) => {
12      22.7 ms     console.log("hhi");
13      1.1 ms     const respuesta = {
14      6.7 ms         argumentosDeEntrada: process.argv,
15      0.8 ms         sistemaOperativo: process.platform,
16      1.4 ms         versionNode: process.version,
17      37.5 ms         memoriaTotalReservada: JSON.stringify(process.memoryUsage()),
18      1.6 ms         pathDeEjecucion: process.execPath,
19      0.7 ms         processId: process.pid,
20      1.5 ms         carpetaDelProyecto: process.cwd(),
21
22     };
23     27.2 ms     return res.json({
24     0.3 ms         pid: process.pid,
25     27.8 ms         ruta: req.baseUrl + req.path,
26     0.8 ms         msg: `HOLA desde puerto ${argv.port}`,
27         data: respuesta,
28     });
29     });
```

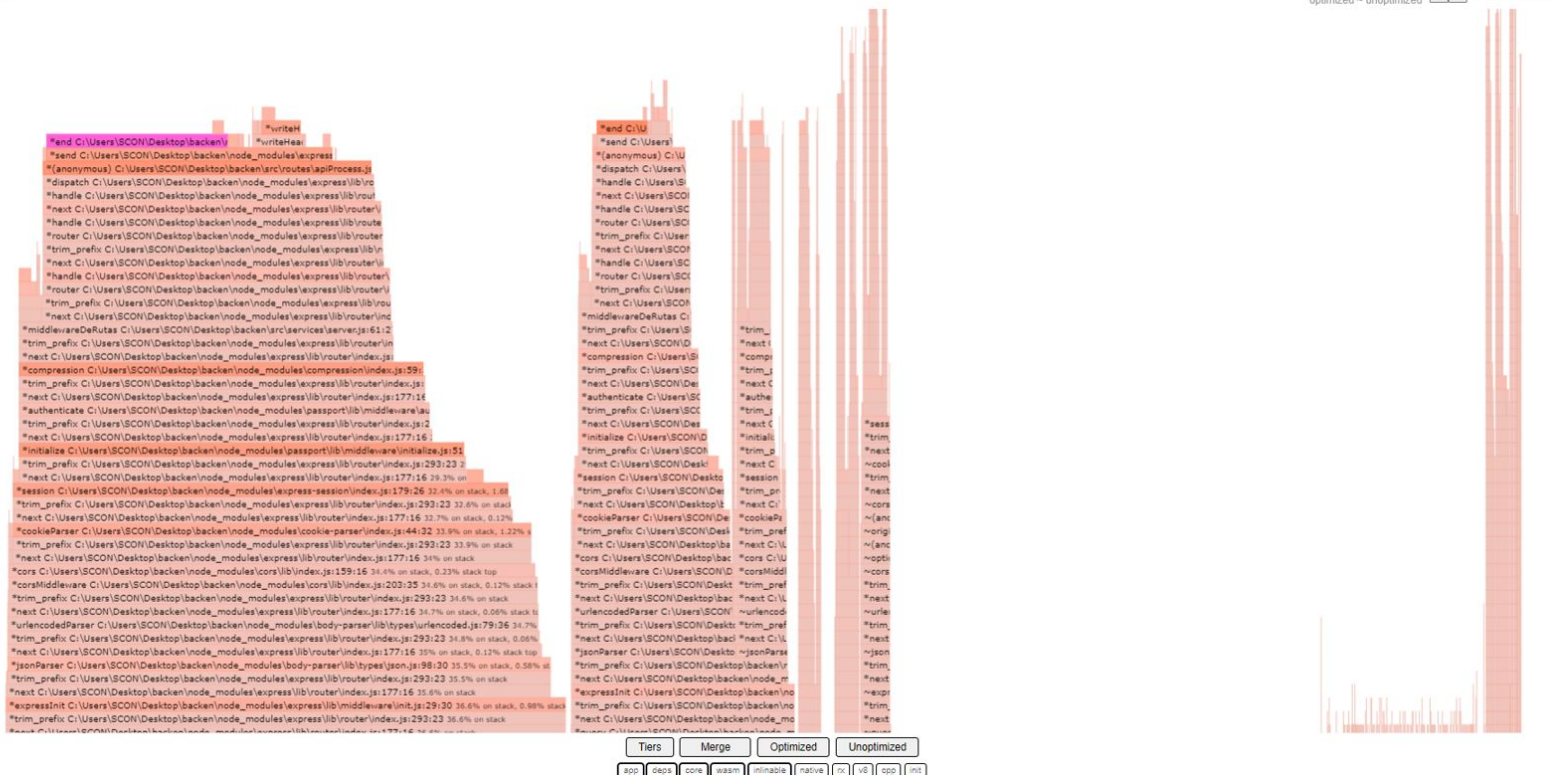

Inspect Sin Console log

```
apiProcess.js x index.js
1      const { fork } = require("child_process");
2      const path = require("path");
3      const { Router } = require("express");
4
5      const routerProcess = Router();
6
7      const yargs = require("yargs/yargs");
8      const { hideBin } = require("yargs/helpers");
9      const argv = yargs(hideBin(process.argv)).argv;
10
11 1 2.2 ms routerProcess.get("/info", (req, res) => {
12 2 0.5 ms   const respuesta = {
13 3 4.8 ms     argumentosDeEntrada: process.argv,
14 4 2.4 ms     sistemaOperativo: process.platform,
15 5 1.6 ms     versionNode: process.version,
16 6 96.9 ms     memoriaTotalReservada: JSON.stringify(process.memoryUsage()),
17 7 2.8 ms     pathDeEjecucion: process.execPath,
18 8 2.5 ms     processId: process.pid,
19 9 1.7 ms     carpetaDelProyecto: process.cwd(),
20 0
21 1 39.4 ms   };
22 2 0.6 ms   return res.json({
23 3 57.8 ms     pid: process.pid,
24 4 1.6 ms     ruta: req.baseUrl + req.path,
25 5 0.2 ms     msg: `HOLA desde puerto ${argv.port}`,
26 6     data: respuesta,
27 7   });
28 8  });
```

0x Con Console

node index.js

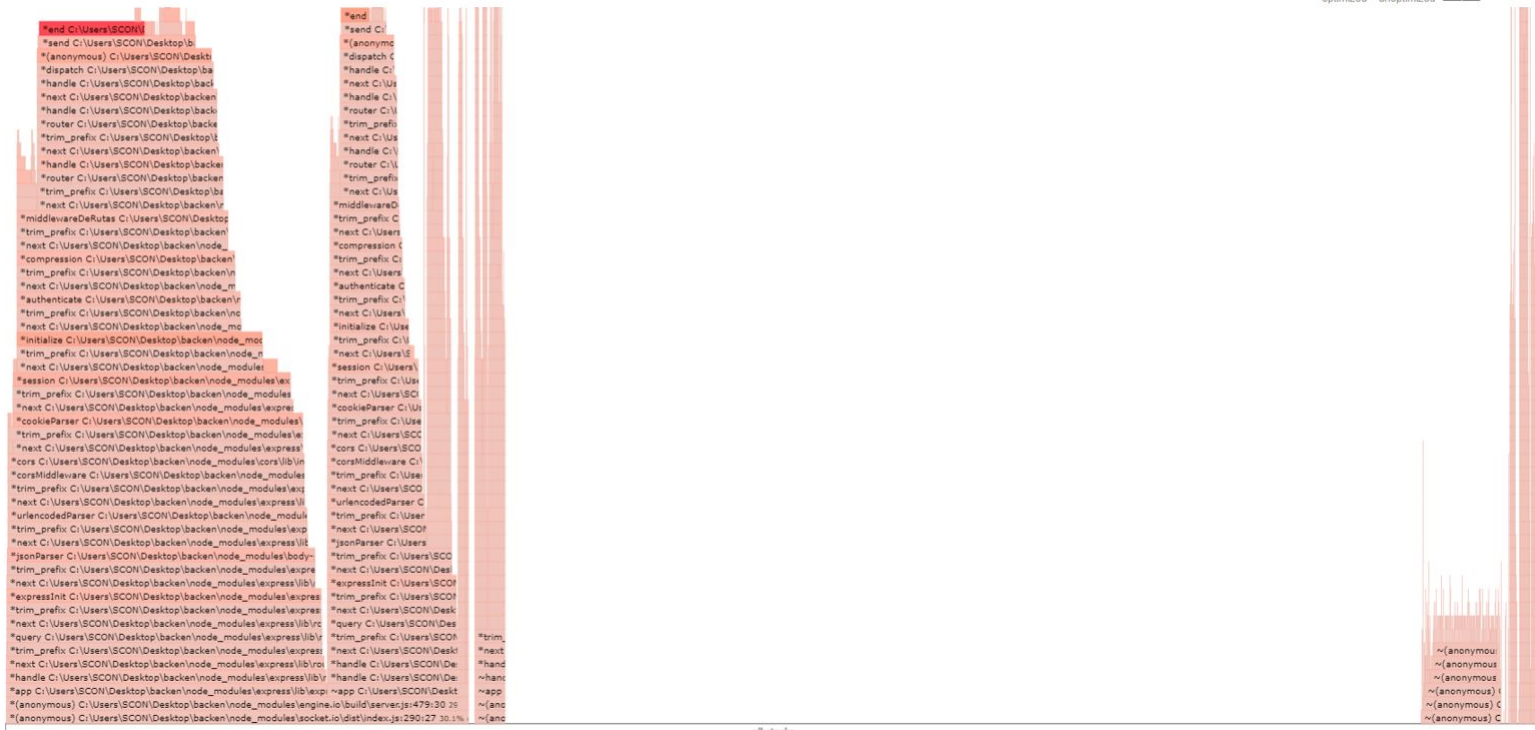
cold hot
* optimized ~ unoptimized



0x Sin Console

node index.js

cold hot
* optimized - unoptimized - + search functions



Conclusión

En conclusión se puede observar por la mayoría de las pruebas que la ejecución con Console log ocasiona más latencia que la ejecución sin Console log. Sin embargo para la prueba del profile la ejecución del Console log ocasionó menos ticks que la que no tiene el Console log.