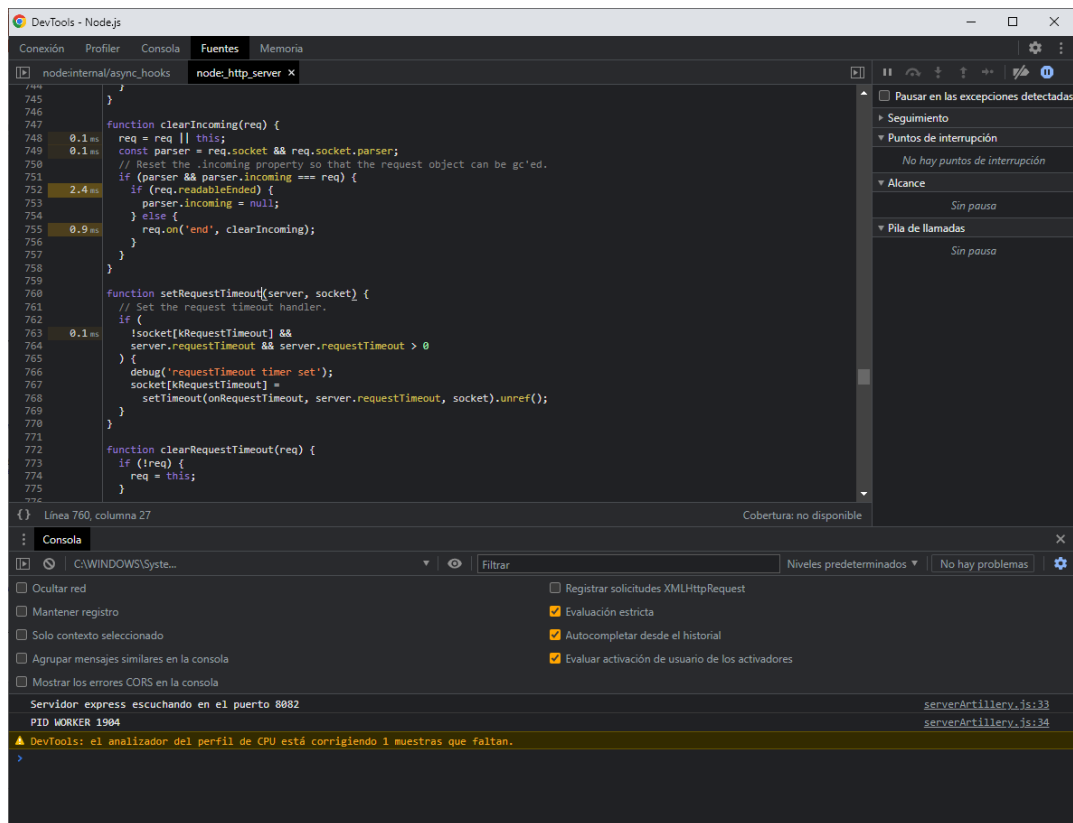


--inspect



DevTools - Node.js

Conexión Profiler Consola Fuentes Memoria

nodeinternal/async_hooks node_http_server X

```
744 }
745 }
746
747 function clearIncoming(req) {
748   req = req || this;
749   const parser = req.socket && req.socket.parser;
750   // Reset the .incoming property so that the request object can be gc'ed.
751   if (parser && parser.incoming === req) {
752     if (req.readableEnded) {
753       parser.incoming = null;
754     } else {
755       req.on('end', clearIncoming);
756     }
757   }
758 }
759
760 function setRequestTimeout(server, socket) {
761   // Set the request timeout handler.
762   if (
763     !socket[kRequestTimeout] &&
764     server.requestTimeout && server.requestTimeout > 0
765   ) {
766     debug('requestTimeout timer set');
767     socket[kRequestTimeout] =
768       setTimeout(onRequestTimeout, server.requestTimeout, socket).unref();
769   }
770 }
771
772 function clearRequestTimeout(req) {
773   if (!req) {
774     req = this;
775   }
776 }
```

{} Línea 760, columna 27 Cobertura: no disponible

Consola

C:\WINDOWS\Syste... Filtrar Niveles predeterminados No hay problemas

☐ Ocultar red ☐ Registrar solicitudes XMLHttpRequest

☐ Mantener registro ☒ Evaluación estricta

☐ Solo contexto seleccionado ☒ Autocompletar desde el historial

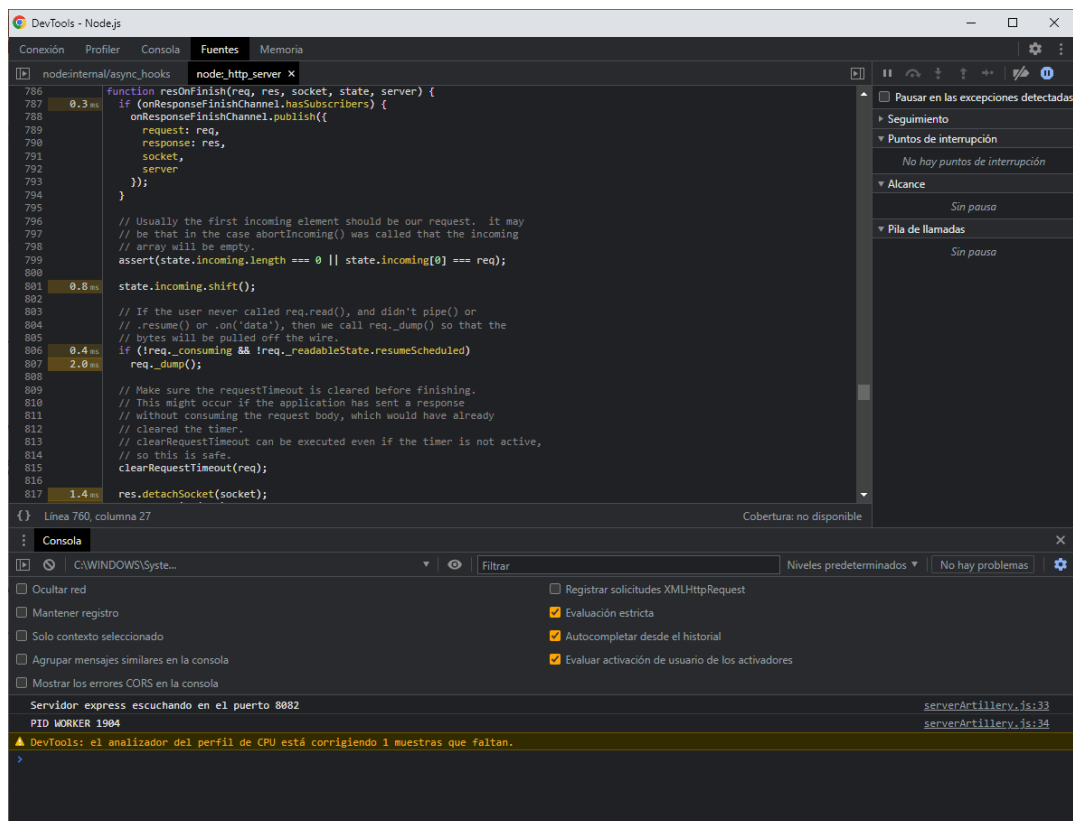
☐ Agrupar mensajes similares en la consola ☒ Evaluar activación de usuario de los activadores

☐ Mostrar los errores CORS en la consola

Servidor express escuchando en el puerto 8082 serverArtillery.js:33

PID WORKER 1904 serverArtillery.js:34

▲ DevTools: el analizador del perfil de CPU está corrigiendo 1 muestras que faltan.



DevTools - Node.js

Conexión Profiler Consola Fuentes Memoria

nodeinternal/async_hooks node_http_server X

```
786 function resOnFinish(req, res, socket, state, server) {
787   if (onResponseFinishChannel.hasSubscribers) {
788     onResponseFinishChannel.publish({
789       request: req,
790       response: res,
791       socket,
792       server
793     });
794   }
795
796   // Usually the first incoming element should be our request. it may
797   // be that in the case abortIncoming() was called that the incoming
798   // array will be empty.
799   assert(state.incoming.length === 0 || state.incoming[0] === req);
800
801   state.incoming.shift();
802
803   // If the user never called req.read(), and didn't pipe() or
804   // .resume() or .on('data'), then we call req._dump() so that the
805   // bytes will be pulled off the wire.
806   if (!req._consuming && !req._readableState.resumeScheduled)
807     req._dump();
808
809   // Make sure the requestTimeout is cleared before finishing.
810   // This might occur if the application has sent a response
811   // without consuming the request body, which would have already
812   // cleared the timer.
813   // clearRequestTimeout can be executed even if the timer is not active,
814   // so this is safe.
815   clearRequestTimeout(req);
816
817   res.detachSocket(socket);
818 }
```

{} Línea 760, columna 27 Cobertura: no disponible

Consola

C:\WINDOWS\Syste... Filtrar Niveles predeterminados No hay problemas

☐ Ocultar red ☐ Registrar solicitudes XMLHttpRequest

☐ Mantener registro ☒ Evaluación estricta

☐ Solo contexto seleccionado ☒ Autocompletar desde el historial

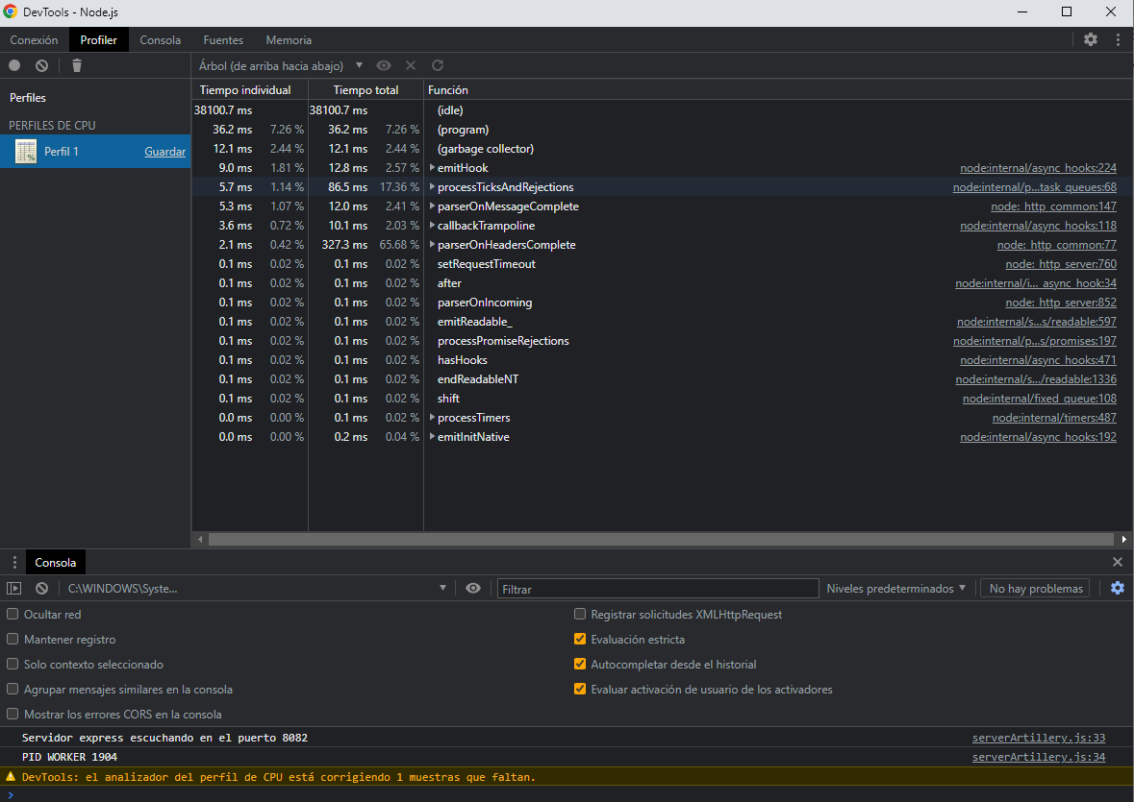
☐ Agrupar mensajes similares en la consola ☒ Evaluar activación de usuario de los activadores

☐ Mostrar los errores CORS en la consola

Servidor express escuchando en el puerto 8082 serverArtillery.js:33

PID WORKER 1904 serverArtillery.js:34

▲ DevTools: el analizador del perfil de CPU está corrigiendo 1 muestras que faltan.



Test con AUTOCANNON

Info debug e info-noddebug (con consola)

flamegraph.html - desafio-14-Loggers-gzip-analysis-performance - Visual Studio Code

EXPLORADOR

- DESAFIO-14-LOGGERS-GZIP-ANALISIS...
 - 9084.0x
 - 16468.0x
 - node_modules
 - src
 - routes
 - apiArtilleryRoutes.js
 - apiRoutes.js
 - CPU-20220802T235422.cpu...
 - getRandom.js
 - .gitignore
 - 2022-08-03_204354.png
 - 2022-08-03_204539.png
 - 2022-08-03_204549.png

TODOS: TREE

- Scan mode: workspace and o...
- desafio-14-Loggers-gzip-a...

ESQUEMA

- LÍNEA DE TIEMPO
- SCRIPTS NPM
 - package.json
 - test: echo "Error no test spe..."
 - start: 0x serverAutocannonjs

flamegraph.html

serverProfiler.js

CPU-20220802T235422.cpuprofile

serverArtillery.js

node

powershell

SALIDA

ProfilingRunning all benchmarks in parallel
Running 20s test @ http://localhost:8082/info-debug
100 connections

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	176 ms	882 ms	1564 ms	1651 ms	876.79 ms	379.59 ms	1746 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	84	84	103	167	111.9	23.62	84
Bytes/Sec	63.3 kB	63.3 kB	77.6 kB	126 kB	84.3 kB	17.8 kB	63.3 kB

Req/Bytes counts sampled once per second.
of samples: 20

2k requests in 20.13s, 1.69 MB read
Running 20s test @ http://localhost:8082/info-nodebug
100 connections

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	195 ms	954 ms	1584 ms	1654 ms	970.55 ms	417.67 ms	1826 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	39	39	100	149	99.8	20.37	39
Bytes/Sec	29.4 kB	29.4 kB	75.3 kB	112 kB	75.2 kB	15.3 kB	29.4 kB

Req/Bytes counts sampled once per second.
of samples: 20

Lin. 1, col. 1 Espacios: 2 UTF-8 LF HTML Go Live Prettier

api/randoms-debug e api/randoms-nodebug (consola)

flamegraph.html - desafio-14-Loggers-gzip-analysis-performance - Visual Studio Code

EXPLORADOR

- DESAFIO-14-LOGGERS-GZIP-ANALISIS...
 - 9084.0x
 - 16468.0x
 - node_modules
 - src
 - routes
 - apiArtilleryRoutes.js
 - apiRoutes.js
 - CPU-20220802T235422.cpu...
 - getRandom.js
 - .gitignore
 - 2022-08-03_204354.png
 - 2022-08-03_204539.png
 - 2022-08-03_204549.png

TODOS: TREE

- Scan mode: workspace and o...
- desafio-14-Loggers-gzip-a...

ESQUEMA

- LÍNEA DE TIEMPO
- SCRIPTS NPM
 - package.json
 - test: echo "Error no test spe..."
 - start: 0x serverAutocannonjs

flamegraph.html

serverProfiler.js

CPU-20220802T235422.cpuprofile

serverArtillery.js

node

powershell

SALIDA

2k requests in 20.15s, 1.5 MB read
Running 20s test @ http://localhost:8082/api/randoms-debug/?cant=50
100 connections

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	92 ms	867 ms	1587 ms	1649 ms	858.13 ms	367.7 ms	1826 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	79	79	101	180	114.15	31.59	79
Bytes/Sec	50.3 kB	50.3 kB	64.2 kB	114 kB	72.5 kB	20.1 kB	50.3 kB

Req/Bytes counts sampled once per second.
of samples: 20

2k requests in 20.17s, 1.45 MB read
Running 20s test @ http://localhost:8082/api/randoms-nodebug/?cant=50
100 connections

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	361 ms	1162 ms	1624 ms	1712 ms	1064.42 ms	371.24 ms	1826 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	46	46	91	115	91.1	14.36	46
Bytes/Sec	29.1 kB	29.1 kB	57.9 kB	73.1 kB	57.9 kB	9.13 kB	29.1 kB

Req/Bytes counts sampled once per second.

Lin. 1, col. 1 Espacios: 2 UTF-8 LF HTML Go Live Prettier

CONCLUSIONES:

Se puede verificar que los tiempos de procesos en las rutas *'/info-nodebug'* y *'api/randoms-nodebug'* son mayores a los de las rutas *'/info-debug'* y *'api/randoms-debug'*, debido a la operación sincrónica bloqueante del `console.log`.