

Chrome Devtools

inside out

Chrome Devtools

inside out

Hi, I'm Katie

opm





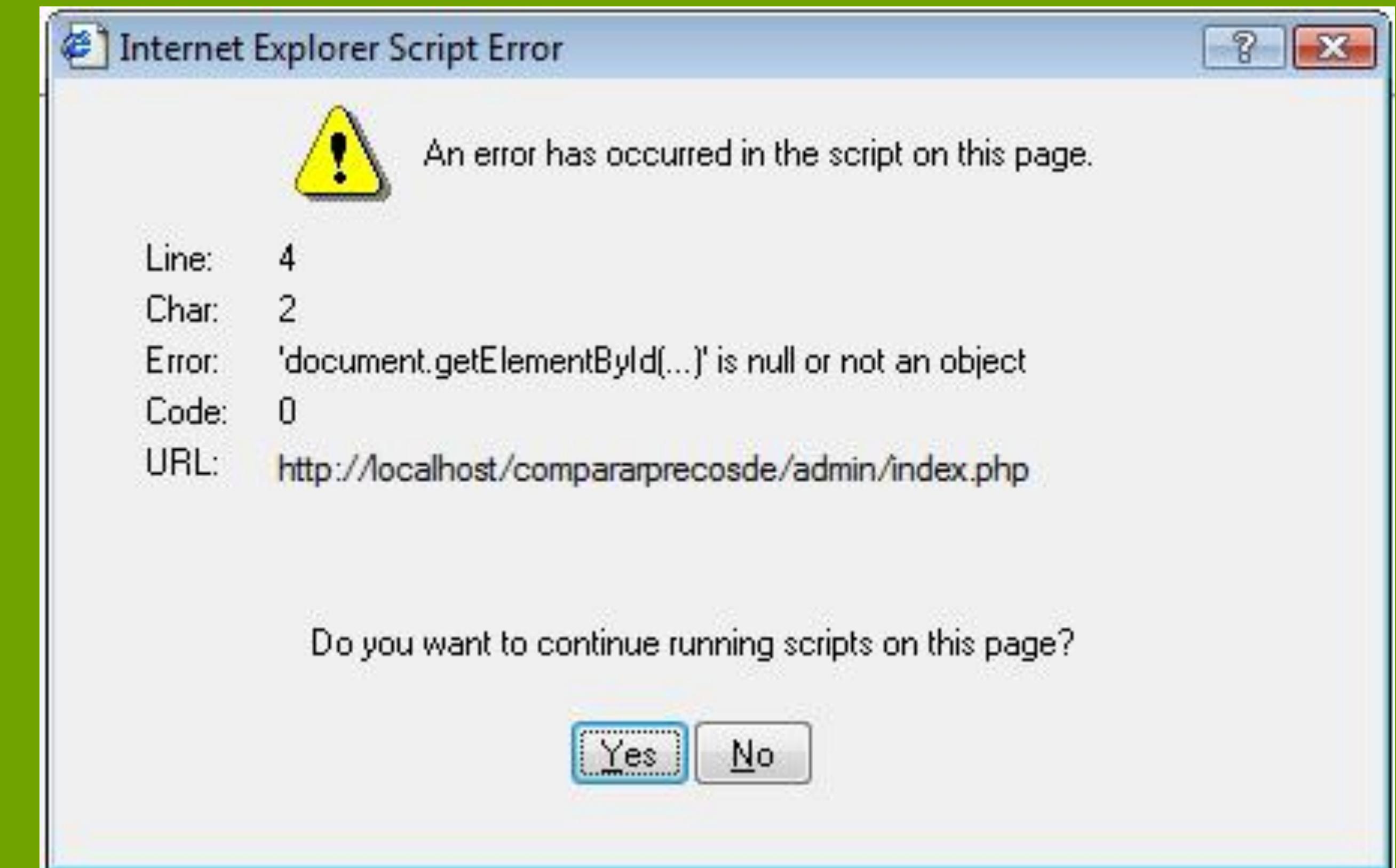
tell me what you think!

content warnings

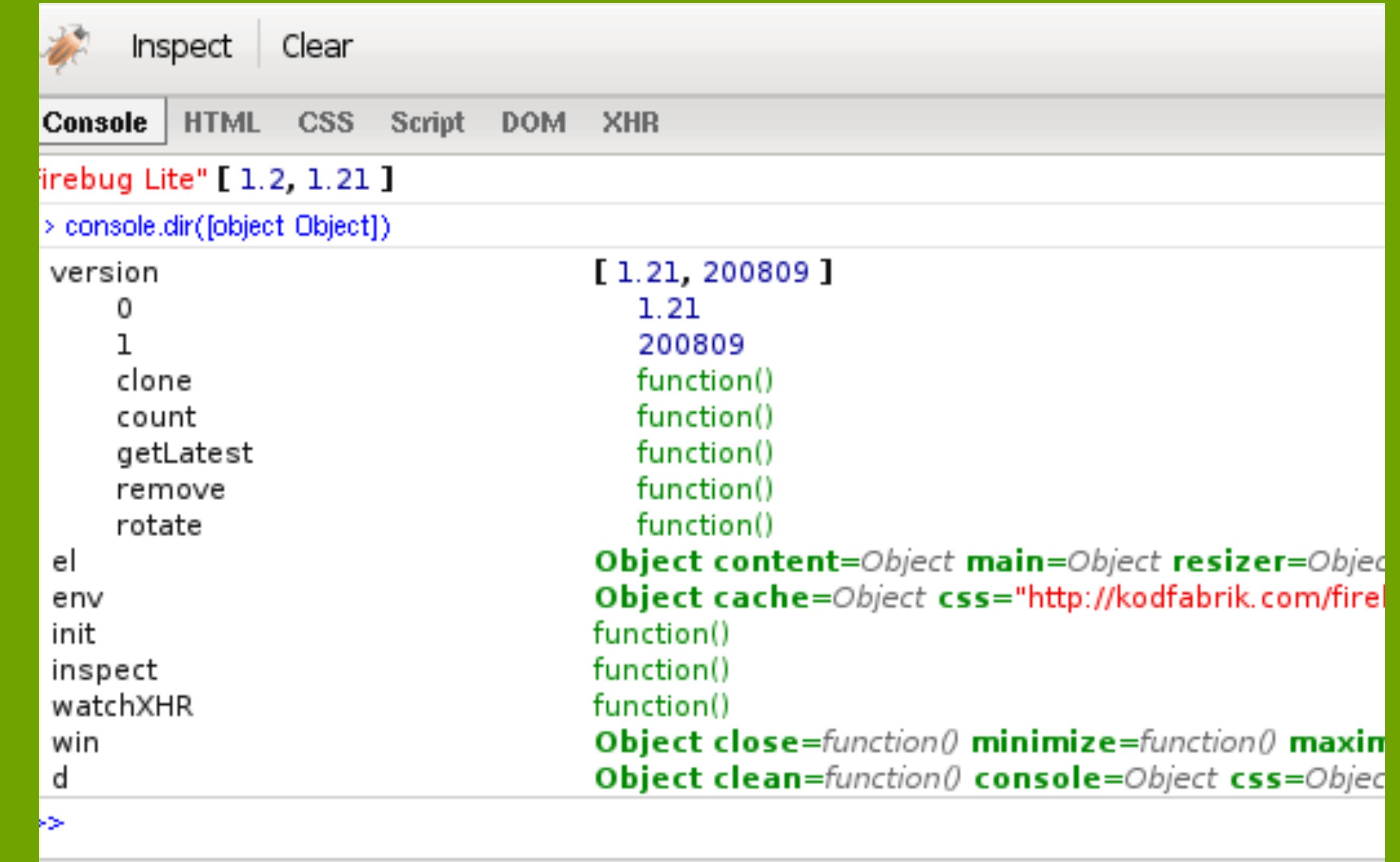


Devtools, 2001

JavaScript errors in Internet Explorer



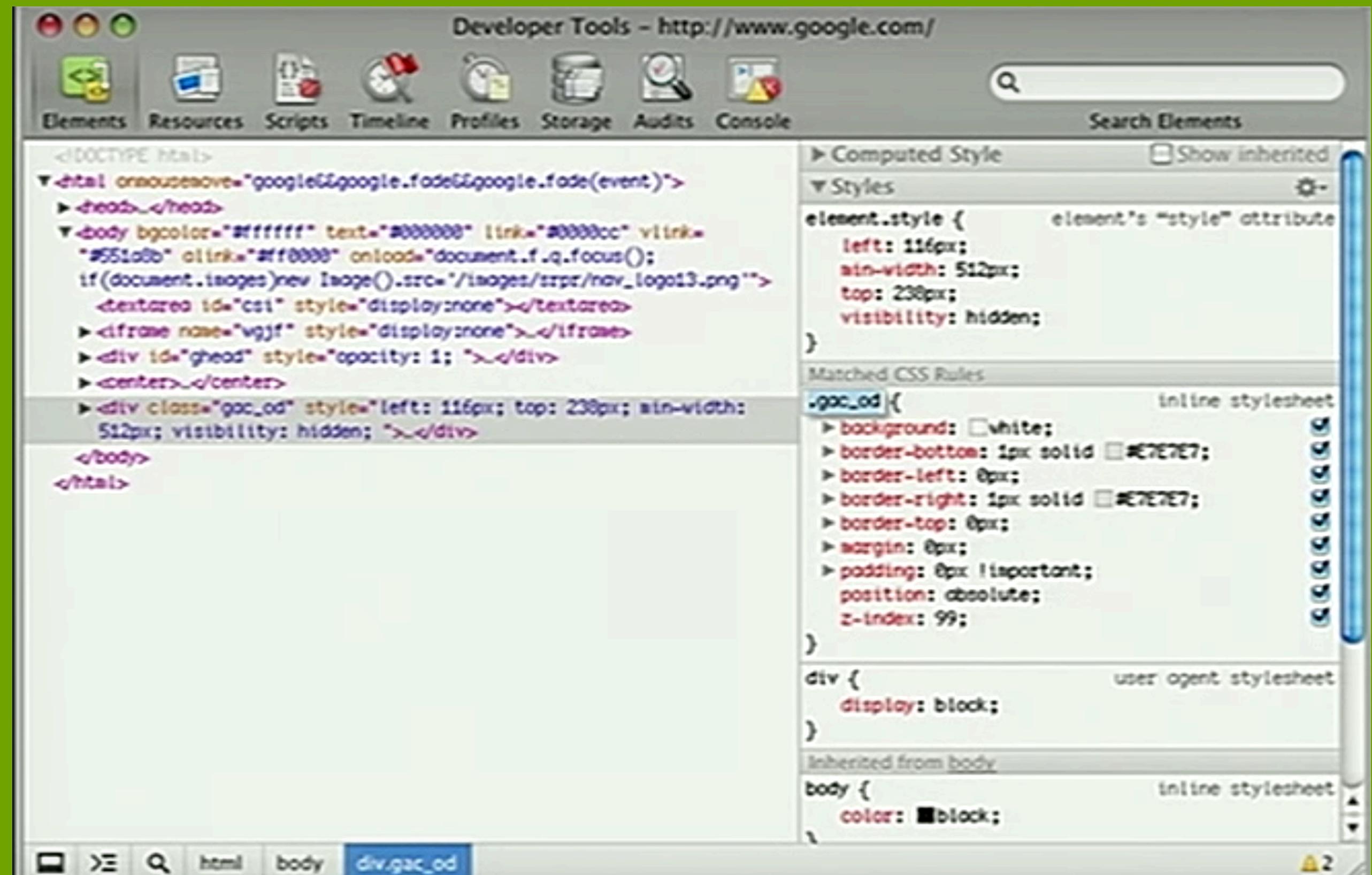
Firebug (2005)



The screenshot shows the Firebug Lite developer toolbar. The title bar includes a bug icon, the text "Inspect | Clear", and tabs for "Console", "HTML", "CSS", "Script", "DOM", and "XHR". The "Console" tab is active. The main area displays the output of the command `> console.dir([object Object])`. The output shows a list of properties and methods, many of which are highlighted in green, indicating they are part of the Firebug API or its extensions. The properties listed include: version, 0, 1, clone, count, getLatest, remove, rotate, el, env, init, inspect, watchXHR, win, d, and several methods: close, minimize, maximize, and clean. The "version" property is annotated with a tooltip: "[1.21, 200809]".

```
firebug Lite" [ 1.2, 1.21 ]
> console.dir([object Object])
version [ 1.21, 200809 ]
  0
  1
  clone
  count
  getLatest
  remove
  rotate
el
env
init
inspect
watchXHR
win
d
Object content=Object main=Object resizer=Object
Object cache=Object css="http://kodfabrik.com/firebug.css"
function()
function()
function()
function()
function()
function()
function()
Object close=function() minimize=function() maximize=function()
Object clean=function() console=Object css=Object
```

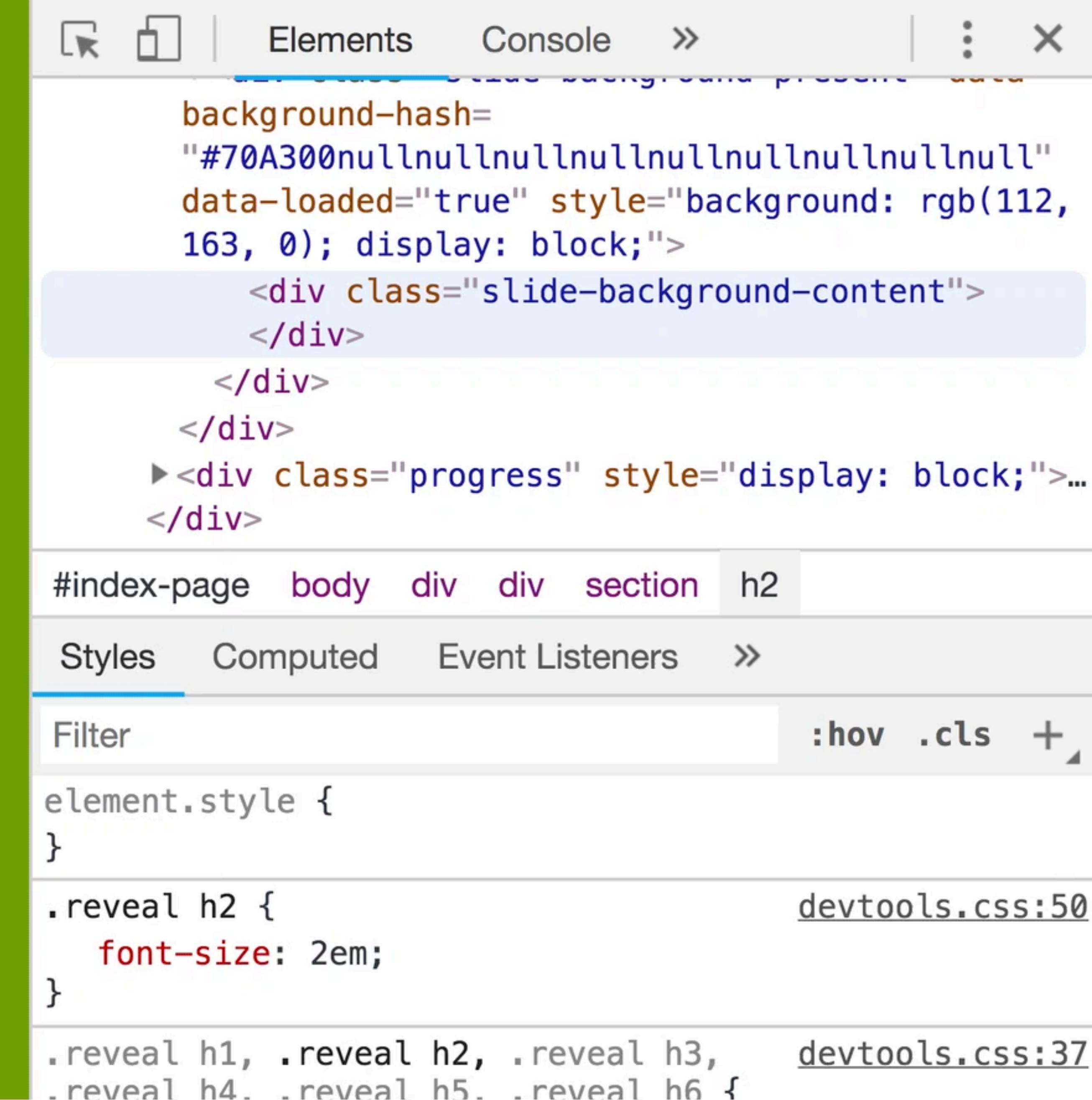
Chrome DevTools (2005)



Elements

Styles tab

Use the styles tab to change CSS properties



The screenshot shows the browser's developer tools open to the 'Elements' tab, specifically the 'Styles' tab under the 'h2' selector. The element selected is a header with the class 'slide-background-content'. The 'Computed' tab is also visible. The CSS properties shown include:

```
background-hash=
"#70A300nullnullnullnullnullnullnullnull"
data-loaded="true" style="background: rgb(112, 163, 0); display: block;">
  <div class="slide-background-content">
    </div>
  </div>
  </div>
  ▶<div class="progress" style="display: block;">
    </div>
```

The 'element.style' section shows:

```
element.style {
```

The '.reveal h2' rule from 'devtools.css:50' shows:

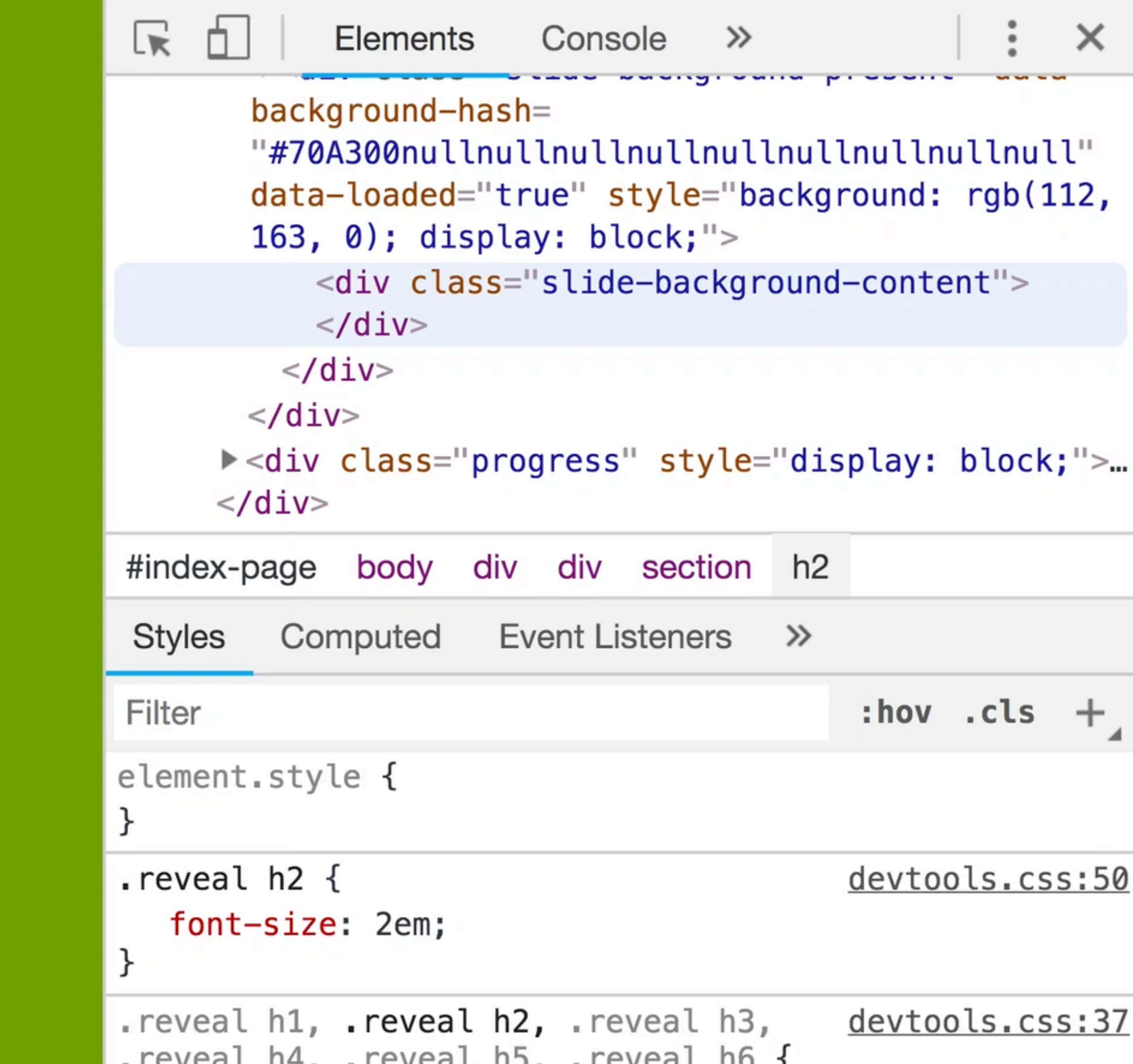
```
.reveal h2 {
  font-size: 2em;
}
```

The '.reveal h1, .reveal h2, .reveal h3, .reveal h4, .reveal h5, .reveal h6' rule from 'devtools.css:37' shows:

```
.reveal h1, .reveal h2, .reveal h3,
.reveal h4, .reveal h5, .reveal h6 {
```

Styles tab

Use the styles tab to change CSS properties



The screenshot shows the browser's developer tools open to the 'Elements' tab, specifically the 'Styles' tab under the 'h2' selector. The element selected is a slide background content div. The CSS properties listed are:

```
background-hash=
"#70A300nullnullnullnullnullnullnullnull"
data-loaded="true" style="background: rgb(112, 163, 0); display: block;">
  <div class="slide-background-content">
    </div>
  </div>
  </div>
  ▶<div class="progress" style="display: block;">
    </div>
```

The 'Styles' tab is active, showing the following CSS rules:

- #index-page { }
- body { }
- div { }
- div { }
- section { }
- h2 { }

Under the 'h2' rule, the following styles are defined:

```
element.style {
}
.reveal h2 {
  font-size: 2em;
}
.reveal h1, .reveal h2, .reveal h3,
.reveal h4, .reveal h5, .reveal h6 {
```

On the right side of the styles panel, there are buttons for ':hov', '.cls', and '+'. The file paths for the styles are indicated as 'devtools.css:50' and 'devtools.css:37'.

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected at the top. Below it, the 'Styles' tab is active. The left pane displays the DOM tree:

```
aria-hidden="true" class="past" style="top: 162.5px; display: block;">...</section>
  <section data-background="#70A300" class="has-light-background present" style="top: 225px; display: block;">
    ...
      <h2>Styles tab</h2> == $0
      ><p>...</p>
    </section>
  </div>
  ><div class="banner">...</div>
  ><div class="backgrounds">
```

The right pane shows the CSS rules for the selected element. The 'element.style' rule contains:

```
element.style {
```

The '.reveal h2' rule from 'devtools.css:50' is expanded, showing:

```
.reveal h2 {
```

With two checked checkboxes:

- font-size: 2em;
- font-family: 'Comic Sans MS';

The bottom of the pane shows other rules:

```
.reveal h1, .reveal h2, .reveal h3, .reveal h4, .reveal h5, .reveal h6 {
```

and

```
devtools.css:37
```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected at the top. Below it, the 'Styles' tab is active. The left pane displays the DOM tree:

```
aria-hidden="true" class="past" style="top: 162.5px; display: block;">...</section>
  <section data-background="#70A300" class="has-light-background present" style="top: 225px; display: block;">
    ...
      <h2>Styles tab</h2> == $0
      ><p>...</p>
    </section>
  </div>
  ><div class="banner">...</div>
  ><div class="backgrounds">
```

The right pane shows the CSS rules for the selected element. The 'element.style' rule contains:

```
element.style {
```

The '.reveal h2' rule from 'devtools.css:50' is expanded, showing:

```
.reveal h2 {
```

With two checked checkboxes:

- font-size: 2em;
- font-family: 'Comic Sans MS';

The bottom of the pane shows other rules:

```
.reveal h1, .reveal h2, .reveal h3, .reveal h4, .reveal h5, .reveal h6 {
```

and

```
devtools.css:37
```

Styles tab

Use the styles tab to change CSS properties



The screenshot shows the browser's developer tools open to the Styles tab. The left pane displays the DOM tree with several `div` elements. The middle pane shows the selected element's style rules. The bottom pane lists the available CSS properties for modification.

DOM Tree:

```
data-loaded="true" style="display: block;"></div>
  <div class="slide-background present" data-background-hash="#70A300nullnullnullnullnullnullnullnullnullnull">
    data-loaded="true" style="background: rgb(112, 163, 0); display: block;">
      <div class="slide-background-content">
        </div> == $0
      </div>
    ...
  ...
  body div div div div.slide-background-content
```

Selected Element:

```
element.style { }
```

Style Rules:

```
.reveal .slide-background-content { reveal.css:1047
  position: absolute;
  width: 100%;
  height: 100%;
  background-position: ▶ 50% 50%;}
```

Styles tab

Use the styles tab to change CSS properties



The screenshot shows the browser's developer tools open to the Styles tab. The left pane displays the DOM tree with several `div` elements. The middle pane shows the selected element's style rules. The bottom pane lists the available CSS properties for modification.

DOM Tree:

```
data-loaded="true" style="display: block;"></div>
  <div class="slide-background present" data-background-hash="#70A300nullnullnullnullnullnullnullnullnullnull">
    data-loaded="true" style="background: rgb(112, 163, 0); display: block;">
      <div class="slide-background-content">
        </div> == $0
      </div>
    ...
  ...
  body div div div div.slide-background-content
```

Selected Element:

```
element.style { }
```

Style Rules:

```
.reveal .slide-background-content { reveal.css:1047
  position: absolute;
  width: 100%;
  height: 100%;
  background-position: ▶ 50% 50%;}
```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the "Styles" tab selected. The top bar has icons for back, forward, and refresh, followed by "Element". Below the bar is a search input field with placeholder text "Select an element in the page to inspect it ⌘ ⌘ C". The main area displays the DOM structure:

```
<div class="slide-background present" data-loaded="true" style="background-color: #70A300; opacity: 1; visibility: visible;">

...


```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the "Styles" tab selected. The top bar has icons for back, forward, and refresh, followed by "Element". Below the bar is a search input field with placeholder text "Select an element in the page to inspect it ⌘ ⌘ C". The main area displays the DOM structure:

```
<div class="slide-background present" data-loaded="true" style="background-color: #70A300; opacity: 1; visibility: visible;">

...


```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected at the top. The element hierarchy on the left lists various HTML elements like sections, divs, and an h2. The 'Styles' tab is active in the bottom navigation bar. On the right, the CSS rules for the current element are listed, including a global rule for element.style and a specific rule for .reveal h2. The file names devtools.css are shown next to the rules.

```
aria-hidden="true" class="past" style="top: 162.5px; display: block;">...</section>
▼<section data-background="#70A300" class="has-light-background present" style="top: 225px; display: block;">
...
  <h2>Styles tab</h2> == $0
  ▶<p>...</p>
  </section>
</div>
▶<div class="banner">...</div>
▼<div class="backgrounds">
#index-page body div div section h2
  Styles Computed Event Listeners »
  Filter :hov .cls +
element.style {
}
.reveal h2 {
  font-size: 2em;
  font-family: 'Comic Sans MS';
}
.reveal h1 .reveal h2 .reveal h3 devtools.css:37
```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected at the top. The element hierarchy on the left lists various HTML elements like sections, divs, and an h2. The 'Styles' tab is active in the bottom navigation bar. On the right, the CSS rules for the current element are listed, starting with 'element.style' and then '.reveal h2'. The rule '.reveal h2' includes 'font-size: 2em;' and 'font-family: 'Comic Sans MS''.

```
aria-hidden="true" class="past" style="top: 162.5px; display: block;">...</section>
▼<section data-background="#70A300" class="has-light-background present" style="top: 225px; display: block;">
...
  <h2>Styles tab</h2> == $0
  ▶<p>...</p>
  </section>
</div>
▶<div class="banner">...</div>
▼<div class="backgrounds">
#index-page body div div section h2
  Styles Computed Event Listeners »
  Filter :hov .cls +
element.style {
}
.reveal h2 {
  font-size: 2em;
  font-family: 'Comic Sans MS';
}
```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected at the top. The element tree on the left shows a section with a background color of #70A300. A specific H2 element is selected, with its class 'styles tab' highlighted. The 'Styles' tab is selected in the bottom navigation bar. The main area displays the following CSS rules:

```
element.style {  
}  
.reveal h2 {  
    font-size: 2em;  
    font-family: 'Comic Sans MS';  
}  
reveal h1 reveal h2 reveal h3 devtools.css:37  
devtools.css:60
```

Styles tab

Use the styles tab to change CSS properties

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected at the top. The element tree on the left shows a section with a background color of #70A300. A specific H2 element is selected, with its class 'styles tab' highlighted. The 'Styles' tab is selected in the bottom navigation bar. The main area displays the following CSS rules:

```
element.style {  
}  
.reveal h2 {  
    font-size: 2em;  
    font-family: 'Comic Sans MS';  
}  
reveal h1 reveal h2 reveal h3 devtools.css:37  
devtools.css:60
```

Intermission



Intermission



Console

console.log

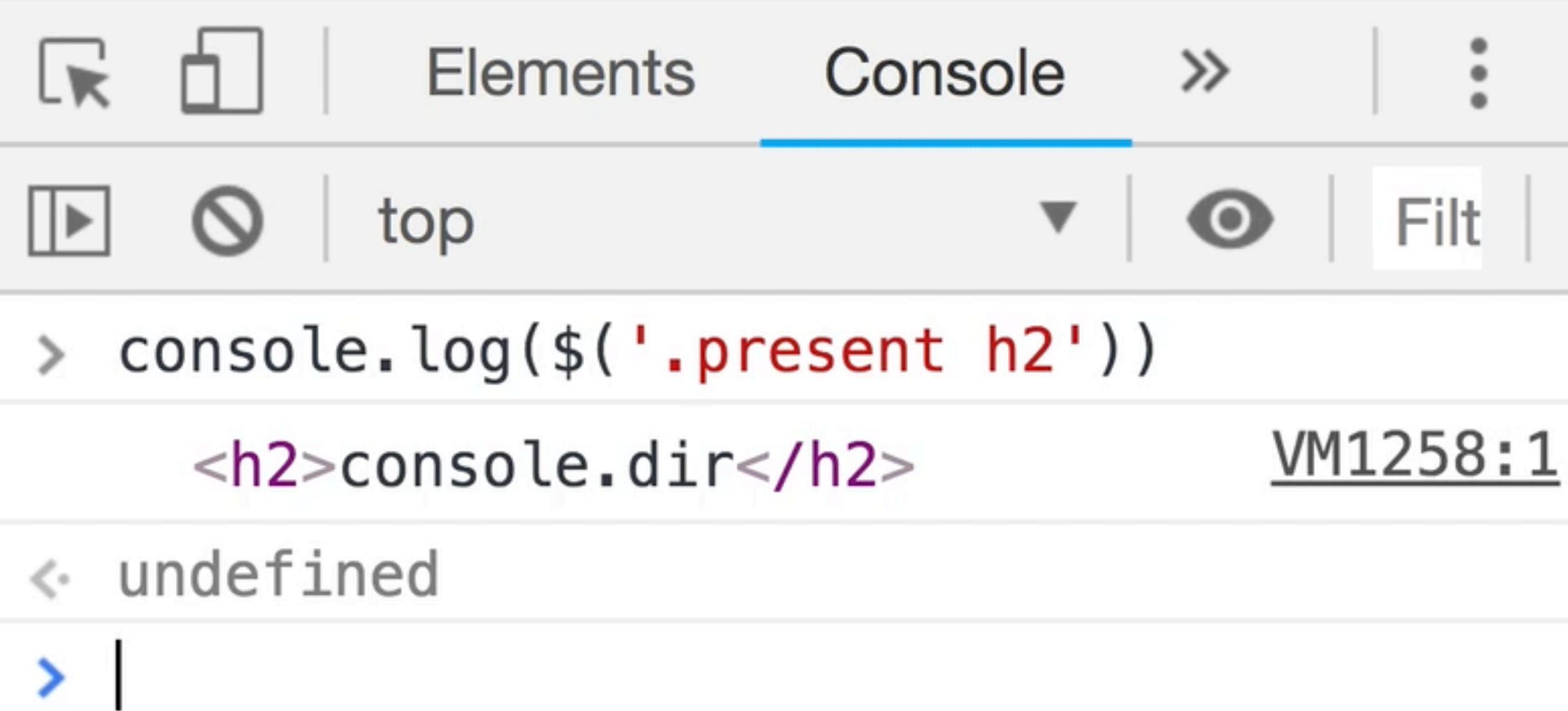
Log messages to the JavaScript console

console.log

Log messages to the JavaScript console

console.dir

Logs object properties

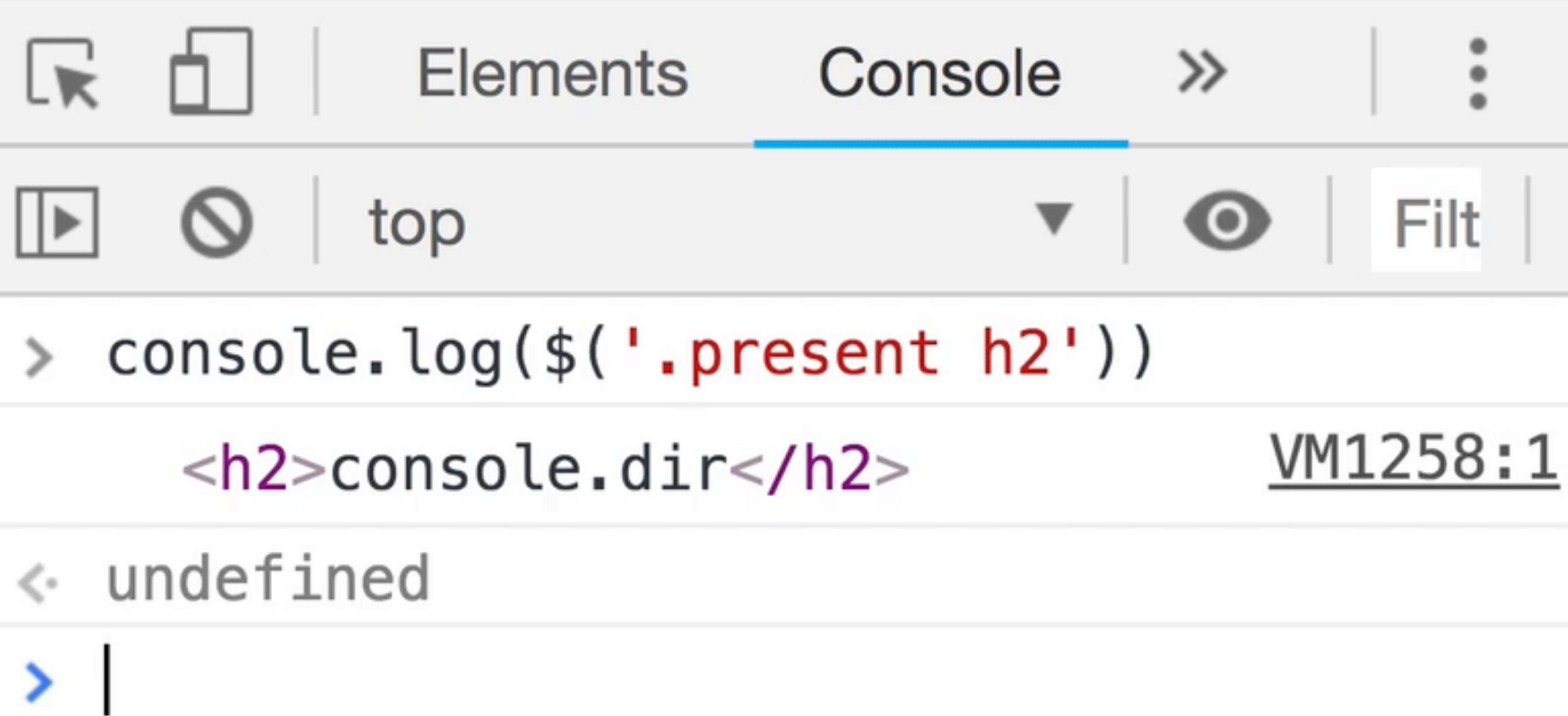


The screenshot shows a browser's developer tools open to the 'Console' tab. At the top, there are icons for back, forward, and refresh, followed by 'Elements' and 'Console'. Below the tabs, there are buttons for play/pause, stop, and filter, with 'top' selected. The main area displays the following log entries:

```
> console.log($('.present h2'))  
    <h2>console.dir</h2> VM1258:1  
< undefined  
> |
```

console.dir

Logs object properties



The screenshot shows a browser's developer tools open to the 'Console' tab. At the top, there are icons for back, forward, and refresh, followed by 'Elements' and 'Console'. Below the tabs, there are buttons for play/pause, stop, and filter, with 'top' selected. The main area displays the following log entries:

```
> console.log($('.present h2'))  
    <h2>console.dir</h2> VM1258:1  
< undefined  
> |
```

console.time

Starts a timer

console.time

Starts a timer

console.memory

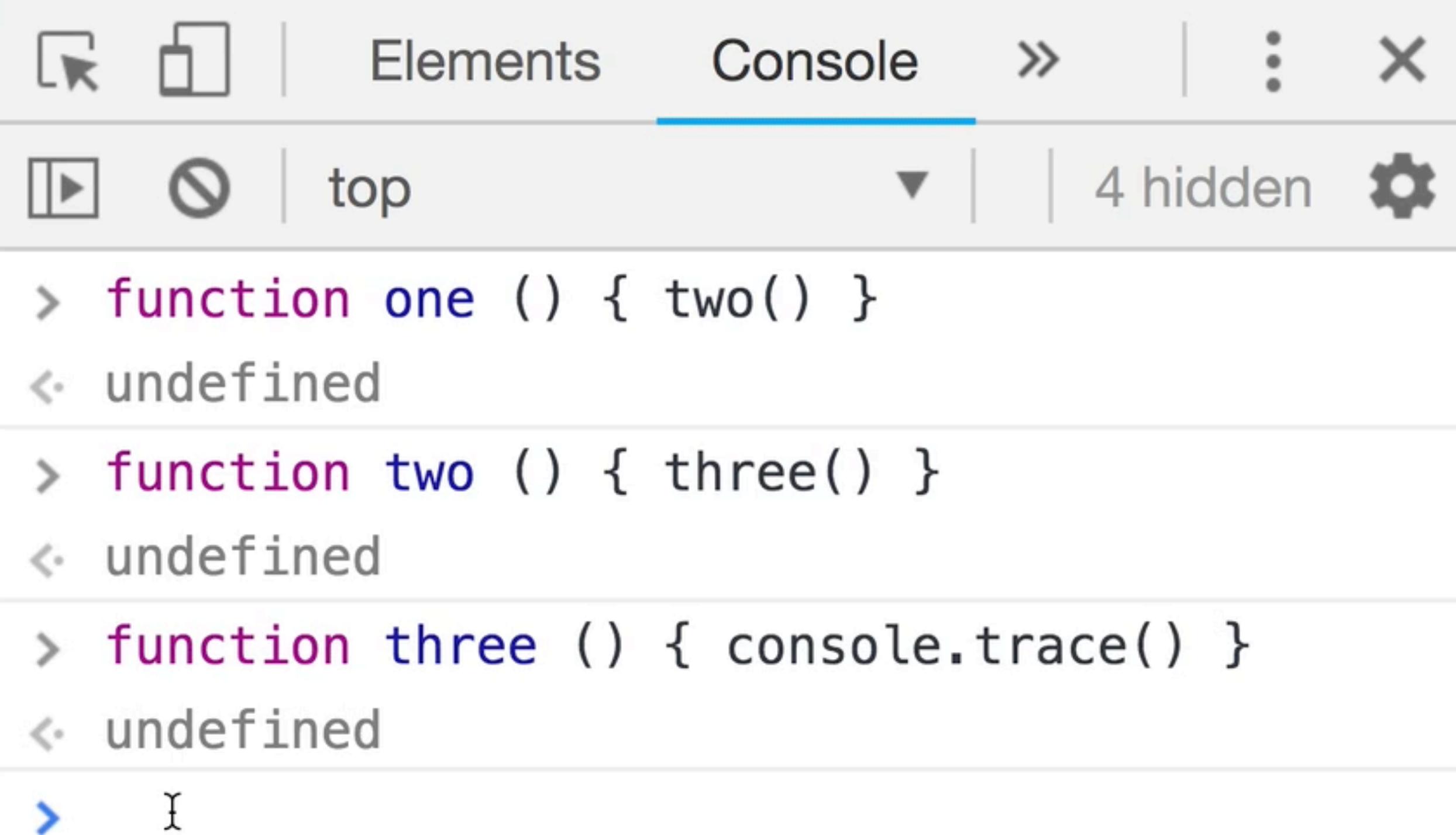
Reports memory heap size

console.memory

Reports memory heap size

console.trace

Logs a stack trace

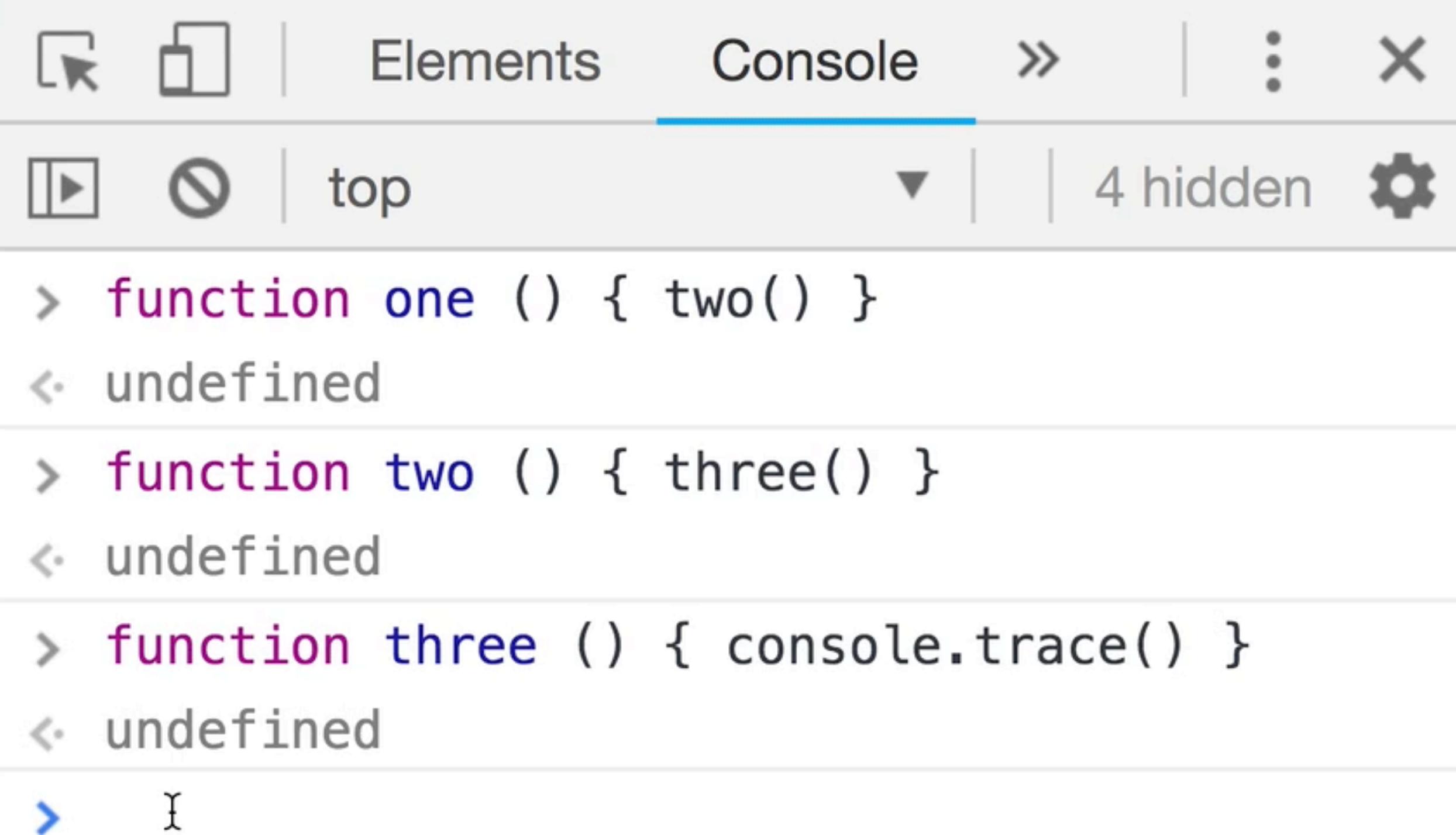


The screenshot shows the browser's developer tools open to the 'Console' tab. At the top, there are icons for back, forward, and search, followed by 'Elements' and 'Console'. Below the tabs, there are buttons for play/pause, stop, and a dropdown set to 'top'. To the right, it says '4 hidden' and has a settings gear icon. The main area displays a stack trace:

```
> function one () { two() }
< undefined
> function two () { three() }
< undefined
> function three () { console.trace() }
< undefined
> I
```

console.trace

Logs a stack trace



The screenshot shows the browser's developer tools open to the 'Console' tab. At the top, there are icons for back, forward, and search, followed by 'Elements' and 'Console'. Below the tabs, there are buttons for play/pause, stop, and a dropdown menu set to 'top'. To the right, it says '4 hidden' and has a settings gear icon. The main area displays a stack trace:

```
> function one () { two() }
< undefined
> function two () { three() }
< undefined
> function three () { console.trace() }
< undefined
> I
```

console.count

Logs an incrementing counter

console.count

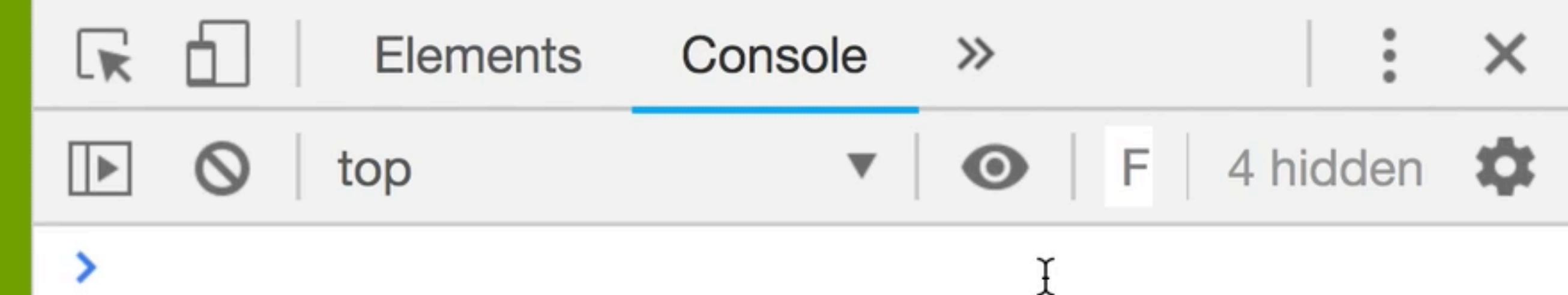
Logs an incrementing counter

Pinned expressions

Keep an eye on values

duck season

rabbit season

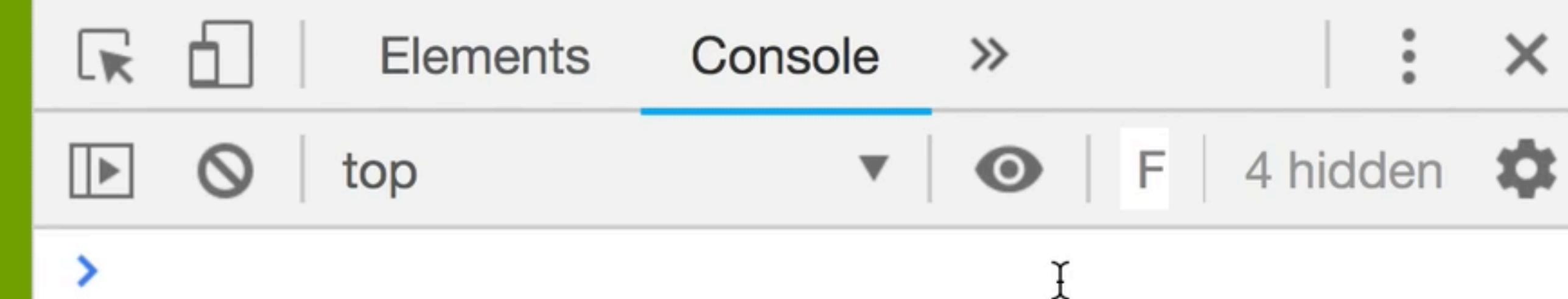


Pinned expressions

Keep an eye on values

duck season

rabbit season



Intermission



Intermission



Sources

Command menu (Cmd / Ctrl + P)

Open files and search commands

The screenshot shows the 'Sources' tab of a browser's developer tools. The file being viewed is '(index)'. The code is a snippet of HTML and CSS. Lines 66 through 81 are visible, containing sections for pinned expressions, sources, and the command menu itself. Line 73 is highlighted with a blue bar. The status bar at the bottom indicates 'Not paused'.

```
<section data-background="#70A300">
  <h2>Pinned expressions</h2>
  <p>Keep an eye on values</p>
  <button onclick="season = 'ducks'">Switch</button>
  <button onclick="season = 'rabbits'">Switch</button>
</section>

<section data-background="#F6511D">
  <h2>Sources</h2>
</section>

<section data-background="#70A300">
  <h2>Command menu (Cmd / Ctrl + P)</h2>
  <p>Open files and search commands</p>
</section>
```

{ } Line 73, Column 3

Scope Watch

Call Stack

Not paused

Breakpoints

Command menu (Cmd / Ctrl + P)

Open files and search commands

The screenshot shows the 'Sources' tab of a browser's developer tools. The file being viewed is '(index)'. The code is a snippet of HTML and CSS. Lines 66 through 81 are visible, containing sections for pinned expressions, sources, and the command menu itself. Line 73 is highlighted with a blue bar. The status bar at the bottom indicates 'Not paused'.

```
<section data-background="#70A300">
  <h2>Pinned expressions</h2>
  <p>Keep an eye on values</p>
  <button onclick="season = 'ducks'">Switch</button>
  <button onclick="season = 'rabbits'">Switch</button>
</section>

<section data-background="#F6511D">
  <h2>Sources</h2>
</section>

<section data-background="#70A300">
  <h2>Command menu (Cmd / Ctrl + P)</h2>
  <p>Open files and search commands</p>
</section>
```

{ } Line 73, Column 3

Scope Watch

Call Stack

Not paused

Breakpoints

Breakpoints

Pause your script

start counter

stop counter

The screenshot shows the browser's developer tools open to the 'Sources' tab. A file named 'counter.js' is selected. The code contains three functions:

```
4 function startCounter () {  
5   count = 0  
6  
7   interval = setInterval(incrementCounter, 1000)  
8 }  
9  
10 function stopCounter () {  
11   clearInterval(interval)  
12 }  
13  
14 function incrementCounter () {  
15   count++  
16   document.querySelectorAll('.counter').forEach(e => e.innerHTML = count)  
17 }  
18
```

A red dot indicates a breakpoint is set on the first line of the 'incrementCounter' function. The status bar at the bottom right shows 'Not paused'.

Breakpoints

Pause your script

start counter

stop counter

The screenshot shows the browser's developer tools open to the 'Sources' tab. A file named 'counter.js' is selected. The code contains three functions:

```
4 function startCounter () {  
5   count = 0  
6  
7   interval = setInterval(incrementCounter, 1000)  
8 }  
9  
10 function stopCounter () {  
11   clearInterval(interval)  
12 }  
13  
14 function incrementCounter () {  
15   count++  
16   document.querySelectorAll('.counter').forEach(e => e.innerHTML = count)  
17 }  
18
```

A red dot indicates a breakpoint is set on the first line of the 'incrementCounter' function. The status bar at the bottom right shows 'Not paused'.

Breakpoints

Pause your script

start counter

stop counter

The screenshot shows the browser's developer tools open to the 'Sources' tab. The file 'counter.js' is selected. The code is as follows:

```
3
4 function startCounter () {
5   count = 0
6
7   interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelector('#counter').innerText = count
17 }
18
```

A red dot on the left margin of line 14 indicates a breakpoint has been set. The status bar at the bottom right of the tools window displays the message "Not paused".

Breakpoints

Pause your script

start counter

stop counter

The screenshot shows the browser's developer tools open to the 'Sources' tab. The file 'counter.js' is selected. The code is as follows:

```
3
4 function startCounter () {
5   count = 0
6
7   interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelector('#counter').innerText = count
17 }
18
```

A red dot on the left margin indicates a breakpoint is set on line 14. The status bar at the bottom right of the tools window displays the message "Not paused".

points

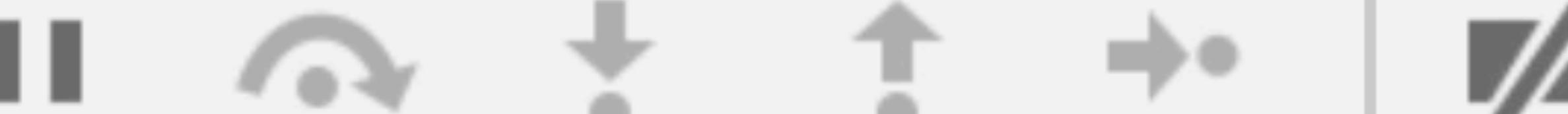
script

stop counter

```
10 function stopCounter()
11   clearInterval(interval)
12 }
13
14 function incrementCounter()
15   count++
16   document.querySelector('#count').innerHTML = count
17 }
18 }
```

{}

Line 7, Column 14



points

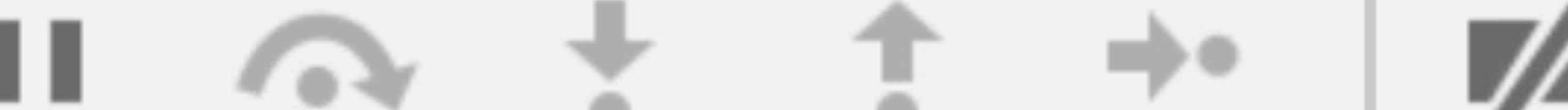
script

stop counter

```
10 function stopCounter()
11   clearInterval(interval)
12 }
13
14 function incrementCounter()
15   count++
16   document.querySelector('#count').innerHTML = count
17 }
18 }
```

{}

Line 7, Column 14



Breakpoints

Pause your script

start counter

stop counter

The screenshot shows the browser's developer tools open to the 'Sources' tab. The file 'counter.js' is selected. The code is as follows:

```
3
4 function startCounter () {
5   count = 0
6
7   interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelectorAll('.counter').forEach(el =>
17     el.textContent = count
18   )
}
```

A red dot on the left margin of line 15 indicates a breakpoint. The status bar at the bottom right of the tools window displays the message "Not paused".

Breakpoints

Pause your script

start counter

stop counter

The screenshot shows the browser's developer tools open to the 'Sources' tab. The file 'counter.js' is selected. The code is as follows:

```
3
4 function startCounter () {
5   count = 0
6
7   interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelectorAll('.counter').forEach(el =>
17     el.textContent = count
18   )
}
```

A red dot on the left margin of line 15 indicates a breakpoint has been set. The status bar at the bottom right of the tools window displays the message "Not paused".

points

our script

3

stop counter

```
7     interval = setInterval(increm  
8 }  
9  
10    function stopCounter () {  
11        clearInterval(interval)  
12    }  
13  
14    function incrementCounter () {  
15        count++  
16        document.querySelectorAll('div').item(count).innerHTML = count  
17    }  
18 }
```

{ } Line 15, Column 3



points

our script

3

stop counter

```
7     interval = setInterval(increm  
8 }  
9  
10    function stopCounter () {  
11        clearInterval(interval)  
12    }  
13  
14    function incrementCounter () {  
15        count++  
16        document.querySelectorAll('div').item(count).innerHTML = count  
17    }  
18 }
```

{ } Line 15, Column 3



Breakpoints

Pause your script

3

start counter

stop counter

```
6
7   interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelectorAll('.counter').forEach(el =>
17     el.textContent = count
18   )
}
```

{ } Line 15, Column 3



ⓘ Paused on breakpoint

▼ Call Stack

▶ incrementCounter
counter.js:15

— setInterval (async)

startCounter counter.js:7

onclick (index):89

Scope	Watch
▼ Local	
▼ this: Window	► LiveReload: LiveRelo... ► Reveal: {VERSION: "3... ► RevealMarkdown: {ini... ► RevealNotes: {open: ... ► alert: f alert() ► applicationCache: Ap... ► atob: f atob()

Breakpoints

Pause your script

3

start counter

stop counter

```
b  
7   interval = setInterval(incrementCounter, 1000)  
8 }  
9  
10 function stopCounter () {  
11   clearInterval(interval)  
12 }  
13  
14 function incrementCounter () {  
15   count++  
16   document.querySelectorAll('.counter').forEach(el  
17 }  
18
```

{ } Line 15, Column 3



ⓘ Paused on breakpoint

▼ Call Stack

▶ incrementCounter
counter.js:15

— setInterval (async)

startCounter counter.js:7

onclick (index):89

Scope	Watch
▼ Local	
▼ this: Window	► LiveReload: LiveRelo... ► Reveal: {VERSION: "3... ► RevealMarkdown: {ini... ► RevealNotes: {open: ... ► alert: f alert() ► applicationCache: Ap... ► atob: f atob()

Breakpoints

Pause your script

3

start counter

stop counter

```
83 </section>
84
85 <section data-background="#70A300">
86   <h2>Breakpoints</h2>
87   <p>Pause your script</p>
88   <p class='counter'></p>
89   <button onclick="startCounter()">start</button>
90   <button onclick="stopCounter()">stop</button>
91 </section>
92
93 <section data-background="#70A300">
94   <h2>Resume script</h2>
95   
```

{ } Line 89, Column 39



Paused on breakpoint

Call Stack

incrementCounter
counter.js:15

setInterval (async)

startCounter counter.js:7

onclick (index):89

Scope	Watch
Local	
this: Window	LiveReload: LiveRelo..., Reveal: {VERSION: "3...", RevealMarkdown: {ini..., RevealNotes: {open: ...}, alert: f alert(), applicationCache: Ap..., atob: f atob()}
LiveReload: LiveRelo..., Reveal: {VERSION: "3...", RevealMarkdown: {ini..., RevealNotes: {open: ...}, alert: f alert(), applicationCache: Ap..., atob: f atob()}	
RevealMarkdown: {ini..., RevealNotes: {open: ...}, alert: f alert(), applicationCache: Ap..., atob: f atob()	
RevealNotes: {open: ...}, alert: f alert(), applicationCache: Ap..., atob: f atob()	
applicationCache: Ap..., atob: f atob()	
atob: f atob()	

Breakpoints

Pause your script

3

start counter

stop counter

```
83 </section>
84
85 <section data-background="#70A300">
86   <h2>Breakpoints</h2>
87   <p>Pause your script</p>
88   <p class='counter'></p>
89   <button onclick="startCounter()">start</button>
90   <button onclick="stopCounter()">stop</button>
91 </section>
92
93 <section data-background="#70A300">
94   <h2>Resume script</h2>
95   
```

{ } Line 89, Column 39



Paused on breakpoint

Call Stack

incrementCounter
counter.js:15

setInterval (async)

startCounter counter.js:7

onclick (index):89

Scope	Watch
Local	
this: Window	LiveReload: LiveRelo... Reveal: {VERSION: "3... RevealMarkdown: {ini... RevealNotes: {open: ... alert: f alert() applicationCache: Ap... atob: f atob()

Resume script



Resumes your script

3

start counter

stop counter

```
6
7     interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11     clearInterval(interval)
12 }
13
14 function incrementCounter () {
15     count++
16     document.querySelectorAll('.counter').forEach(el =>
17         el.textContent = count
18     )
}
```

{ } Line 15, Column 3

The screenshot shows a browser developer tools debugger interface. At the top, there's a toolbar with icons for play, step, and breakpoints. Below it, a message says "Paused on breakpoint". The call stack shows the current execution path: "incrementCounter" at "counter.js:15" (which triggered the breakpoint), followed by "setInterval (async)", "startCounter" at "counter.js:7", and "onclick" at "(index):98". A "Breakpoints" section is also visible. On the right, there are "Scope" and "Watch" tabs, with "Scope" currently selected. The "Local" scope pane lists various variables, including "this: Window", "LiveReload: Liv...", "Reveal: {VERSI0...", "RevealMarkdown:...", "RevealNotes: {o...", "alert: f alert()", "applicationCach...", "atob: f atob()", and "blur: f (...)".

Resume script



Resumes your script

start counter

stop counter



```
5 count = 0
6
7 interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelectorAll('.counter').forEach(el
17 }
```

{ } Line 15, Column 3



Scope Watch

▼ Call Stack

Not paused

Not paused

▶ Breakpoints

▶ XHR/fetch Breakpoints

▶ DOM Breakpoints

▶ Global Listeners

▶ Event Listener Breakpoints

Resume script



Resumes your script

start counter

stop counter



```
5 count = 0
6
7 interval = setInterval(incrementCounter, 1000)
8 }
9
10 function stopCounter () {
11   clearInterval(interval)
12 }
13
14 function incrementCounter () {
15   count++
16   document.querySelectorAll('.counter').forEach(el
17 }
```

{ } Line 15, Column 3



Scope Watch

▼ Call Stack

Not paused

Not paused

▶ Breakpoints

▶ XHR/fetch Breakpoints

▶ DOM Breakpoints

▶ Global Listeners

▶ Event Listener Breakpoints

Step over



Advance script to next line

add verse

```
2
3 function addVerse () {
4     printVerse(count)
5     count = count - 1
6 }
7
8 function printVerse (count) {
9     const elem = document.createElement('div')
10    const text = document.createTextNode(
11        `${count} bottles of beer on the wall,
12        ${count} bottles of beer,
13        you take one down and pass it around,
14        ${count - 1} bottles of beer on the wall`
```

{ } Line 4, Column 3



Scope Watch

▼ Call Stack

Not paused

▼ Breakpoints

No breakpoints

► XHR/fetch Breakpoints

► DOM Breakpoints

► Other Breakpoints

Step over



Advance script to next line

add verse

```
2
3 function addVerse () {
4     printVerse(count)
5     count = count - 1
6 }
7
8 function printVerse (count) {
9     const elem = document.createElement('div')
10    const text = document.createTextNode(
11        `${count} bottles of beer on the wall,
12        ${count} bottles of beer,
13        you take one down and pass it around,
14        ${count - 1} bottles of beer on the wall`
```

{ } Line 4, Column 3



Scope Watch

▼ Call Stack

Not paused

▼ Breakpoints

No breakpoints

► XHR/fetch Breakpoints

► DOM Breakpoints

► Other Breakpoints

Step over



Advance script to next line

add verse

```
3 function addVerse () {  
4     printVerse(count)  
5     count = count - 1  
6 }  
7  
8 function printVerse (count) {  
9     const elem = document.createElement('div')  
10    const text = document.createTextNode(  
11        `${count} bottles of beer on the wall,  
12        ${count} bottles of beer,  
13        you take one down and pass it around,  
14        ${count - 1} bottles of beer on the wall`  
15    elem.appendChild(text)
```

{ } Line 4, Column 3

|| ⏪ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ Scope Watch

▼ Call Stack

Not paused

▼ Breakpoints

No breakpoints

▶ XHR/fetch Breakpoints

▶ DOM Breakpoints

▶ Other Breakpoints

Step over



Advance script to next line

add verse

```
3 function addVerse () {  
4   printVerse(count)  
5   count = count - 1  
6 }  
7  
8 function printVerse (count) {  
9   const elem = document.createElement('div')  
10  const text = document.createTextNode(  
11    `${count} bottles of beer on the wall,  
12    ${count} bottles of beer,  
13    you take one down and pass it around,  
14    ${count - 1} bottles of beer on the wall`  
15  elem.appendChild(text)
```

{ } Line 4, Column 3



Scope Watch

▼ Call Stack

Not paused

▼ Breakpoints

No breakpoints

► XHR/fetch Breakpoints

► DOM Breakpoints

► Other Breakpoints

Step into



Pause script on first line of next function

add verse

```
2
3 function addVerse () {
4     printVerse(count)
5     count = count - 1
6 }
7
8 function printVerse (count) {
9     const elem = document.createElement('div')
10    const text = document.createTextNode(
11        `${count} bottles of beer on the wall,
12        ${count} bottles of beer,
13        you take one down and pass it around,
14        ${count - 1} bottles of beer on the wall`)
```

{ } Line 5, Column 3

Scope Watch Not paused

▼ Call Stack Not paused

▼ Breakpoints No breakpoints

- ▶ XHR/fetch Breakpoints
- ▶ DOM Breakpoints
- ▶ Object Breakpoints

Step into



Pause script on first line of next function

add verse

```
2
3 function addVerse () {
4     printVerse(count)
5     count = count - 1
6 }
7
8 function printVerse (count) {
9     const elem = document.createElement('div')
10    const text = document.createTextNode(
11        `${count} bottles of beer on the wall,
12        ${count} bottles of beer,
13        you take one down and pass it around,
14        ${count - 1} bottles of beer on the wall`)
```

{ } Line 5, Column 3

Scope Watch Not paused

▼ Call Stack Not paused

▼ Breakpoints No breakpoints

- ▶ XHR/fetch Breakpoints
- ▶ DOM Breakpoints
- ▶ Object Breakpoints

Pause on exception



Pause script before exceptions

cause error

```
1 function causeError () {  
2   isnDefined.alsoIsntDefined()  
3 }  
4
```

{ } Line 2, Column 3

Scope Watch Not paused

▼ Call Stack Not paused

▼ Breakpoints

song.js:4
printVerse(count)

▶ XHR/fetch Breakpoints

▶ DOM Breakpoints

Pause on exception



Pause script before exceptions

cause error

```
1 function causeError () {  
2   isntDefined.alsoIsntDefined()  
3 }  
4
```

{ } Line 2, Column 3

|| ⏪ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺

Scope Watch

▼ Call Stack

Not paused

▼ Breakpoints

song.js:4
printVerse(count)

▶ XHR/fetch Breakpoints

▶ DOM Breakpoints

Intermission



Intermission



Network

[Join](#)[Log In](#) Search packages[Search](#)

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

@katie_fenn

Hit ⌘ R to reload and capture filmstrip.

[Join](#)[Log In](#) Search packages[Search](#)

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

The screenshot shows the Network tab of the Chrome DevTools. At the top, there are icons for refresh, copy, elements, console, sources, network (which is highlighted in blue), and performance. Below that is a toolbar with a red circle, a black circle with a slash, a video camera icon, a filter icon, and a search icon. To the right of the toolbar are buttons for 'View' (grid, list, grouped), 'Group by frame', and 'Preserve log'. Further down are 'Filter' and 'Hide data URLs' checkboxes, and tabs for 'All' (selected), XHR, JS, CSS, and Img. A large green circle highlights the 'Manifest' tab, which is currently selected. Other tabs visible include 'Other'. A message at the bottom of the Network tab area says 'Hit ⌘ R to reload and capture'.

[Join](#)[Log In](#) Search packages[Search](#)

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

The screenshot shows the Network tab of the Chrome DevTools. At the top, there are icons for refresh, copy, elements, console, sources, network (which is highlighted), and performance. Below that is a toolbar with a red circle, a lock icon, a video camera icon, a magnifying glass, and a search icon. To the right of the toolbar are buttons for 'View' (grid, list, grouped), 'Group by frame', and 'Preserve log'. Further down are 'Filter' and 'Hide data URLs' checkboxes, and tabs for All, XHR, JS, CSS, and Img. A large green circle highlights the 'All' tab. Below the toolbar, the main area shows a list of network requests. One request for 'manifest' is circled in green. Other items like 'offline', 'online', 'offline.manifest', and 'offline.manifest.map' are also visible. A message at the bottom says 'Hit ⌘ R to reload and capture'.

@katie_fenn

Group by frame

 Preserve log Disable cache Offline Fast 3G

URLs

All

XHR

JS

CSS

Img

Media

Font

Doc

WS

Manifest Other

Search packages



Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

@katie_fenn

Group by frame Preserve log Disable cache Offline Fast 3G

URLs

All

XHR

JS

CSS

Img

Media

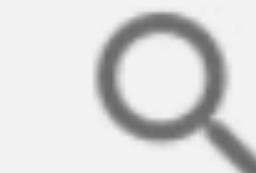
Font

Doc

WS

Manifest

Other

 Search packages

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

@katie_fenn

@katie_fenn

@katie_fenn



Join

Log In

Search packages

Search

Build amazing things

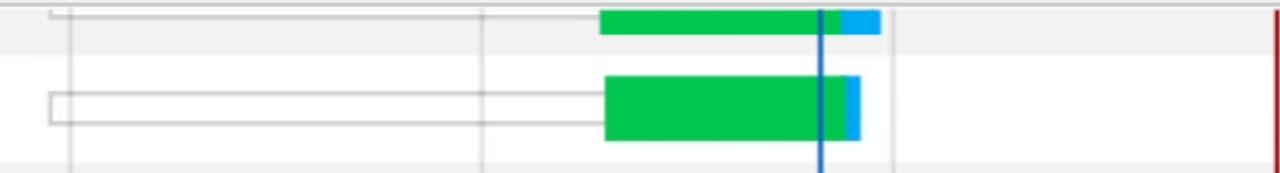
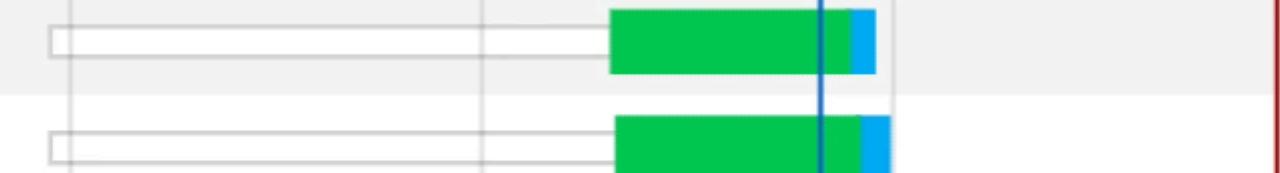
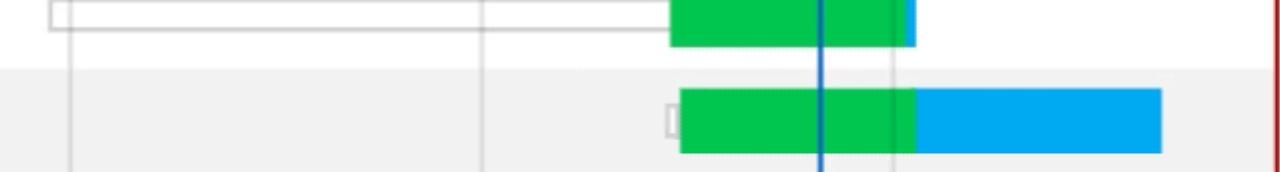
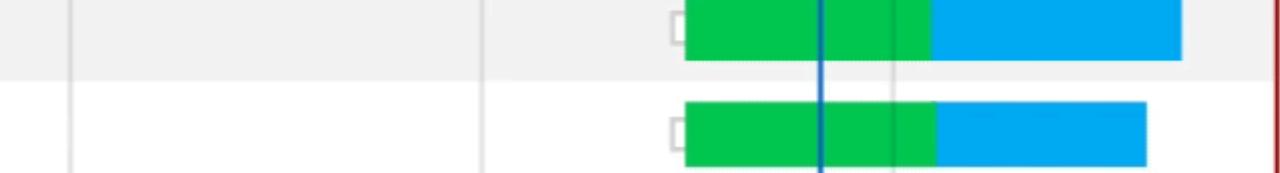
Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

@katie_fenn

Filter Hide data URLs **All** XHR JS CSS Img Media Font Doc WS Manifest Other

Fetching frames...

Name	Status	Size	Time	Waterfall
cjsdu9zpy00pnph74v954fiy4...	200	1.8 KB	619 ms	
cjsdu9zr3241ju674paffp7fz-n...	200	2.2 KB	640 ms	
cjsdu9zk6241qsp74fgw3pn0...	200	3.5 KB	672 ms	
cjsdu9zpy00poph74ndodeid...	200	3.5 KB	681 ms	
cjsdu9zxt241tsp74k3r5xpyr...	200	2.8 KB	597 ms	
pxiByp8kv8JHgFVrLGT9Z1xl...	200	10.3 KB	1.18 s	
6xKydSBYKcSV-LCoeQqfX1...	200	12.6 KB	1.26 s	
N0bX2SIFPv1weGeLZDtgJv7...	200	10.8 KB	1.20 s	
pxiEyp8kv8JHgFVrJJfedw.ttf	200	10.3 KB	1.21 s	
pxiByp8kv8JHgFVrLEj6Z1xlE...	200	10.3 KB	1.21 s	
pxiByp8kv8JHgFVrLCz7Z1xl...	200	10.2 KB	1.13 s	
QldXNThLqRwH-OJ1UHjlKG...	200	25.0 KB	1.42 s	
QldKNThLqRwH-OJ1UHjlKG...	200	21.6 KB	1.40 s	
6xK3dSBYKcSV-LCoeQqfX1...	200	12.7 KB	1.28 s	
analytics.js	307	0 B	105 ms	
cjsdu9td12400sp74em5xcphj...	200	2.3 KB	752 ms	



Join

Log In

Search packages

Search

Build amazing things

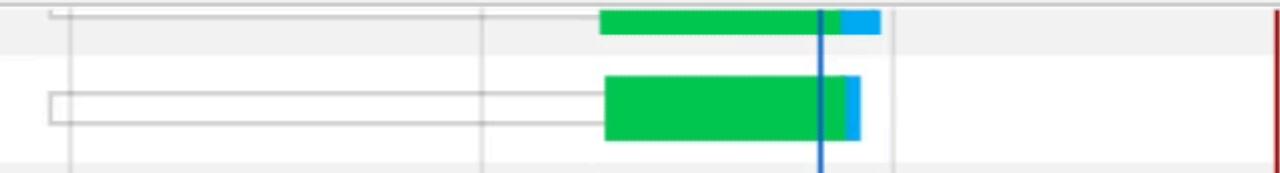
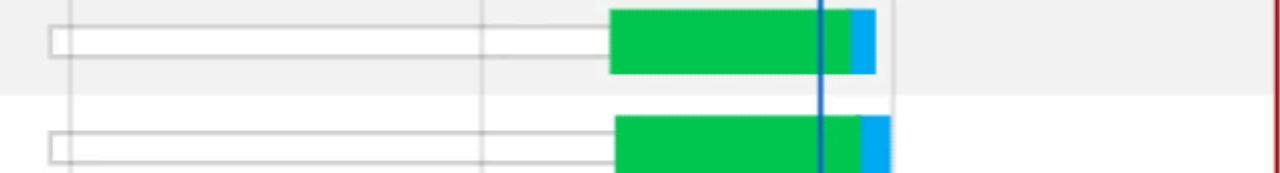
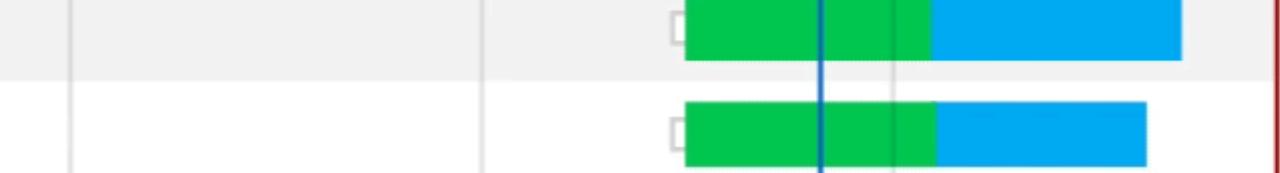
Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

@katie_fenn

Filter Hide data URLs **All** XHR JS CSS Img Media Font Doc WS Manifest Other

Fetching frames...

Name	Status	Size	Time	Waterfall
cjsdu9zpy00pnph74v954fiy4...	200	1.8 KB	619 ms	
cjsdu9zr3241ju674paffp7fz-n...	200	2.2 KB	640 ms	
cjsdu9zk6241qsp74fgw3pn0...	200	3.5 KB	672 ms	
cjsdu9zpy00poph74ndodeid...	200	3.5 KB	681 ms	
cjsdu9zxt241tsp74k3r5xpyr...	200	2.8 KB	597 ms	
pxiByp8kv8JHgFVrLGT9Z1xl...	200	10.3 KB	1.18 s	
6xKydSBYKcSV-LCoeQqfX1...	200	12.6 KB	1.26 s	
N0bX2SIFPv1weGeLZDtgJv7...	200	10.8 KB	1.20 s	
pxiEyp8kv8JHgFVrJJfedw.ttf	200	10.3 KB	1.21 s	
pxiByp8kv8JHgFVrLEj6Z1xlE...	200	10.3 KB	1.21 s	
pxiByp8kv8JHgFVrLCz7Z1xl...	200	10.2 KB	1.13 s	
QldXNThLqRwH-OJ1UHjlKG...	200	25.0 KB	1.42 s	
QldKNThLqRwH-OJ1UHjlKG...	200	21.6 KB	1.40 s	
6xK3dSBYKcSV-LCoeQqfX1...	200	12.7 KB	1.28 s	
analytics.js	307	0 B	105 ms	
cjsdu9td12400sp74em5xcphj...	200	2.3 KB	752 ms	

Search packages

Search

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

See plans

@katie_fenn

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

Recording frames...

Recording network activity...

Perform a request or hit ⌘ R to record the reload.



Join

Log In

Search packages

Search

Build amazing things

Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

Hit ⌘ R to reload and capture filmstrip.

Name	Status	Size	Time	Waterfall
cjsdu9zpy00poph74ndodeid...	200	1.0 KB	200 ms	
cjsdu9zr3241ju674paffp7fz-n...	200	2.3 KB	187 ms	
cjsdu9zk6241qsp74fgw3pn0...	200	3.5 KB	187 ms	
cjsdu9zpy00poph74ndodeid...	200	3.5 KB	364 ms	
5326678.js	(block)	0 B	12 ms	
cjsdu9zxt241tsp74k3r5xpyr...	200	2.8 KB	64 ms	
cjsdu9td12400sp74em5xcphj...	200	2.4 KB	238 ms	
cjsdu9rpp243yr174bmz8nu5...	200	6.1 KB	315 ms	
cjsdqox9k0kgel1beo12x4x6e...	200	198 B	260 ms	
pxiByp8kv8JHgFVrLGT9Z1xl...	200	10.4 KB	1.87 s	
6xKydSBYKcSV-LCoeQqfX1...	200	12.7 KB	533 ms	
N0bX2SIFPv1weGeLZDtgJv7...	200	10.8 KB	3.00 s	
pxiEyp8kv8JHgFVrJJfedw.ttf	200	10.3 KB	2.86 s	
pxiByp8kv8JHgFVrLEj6Z1xE...	200	10.3 KB	2.53 s	
pxiByp8kv8JHgFVrLCz7Z1xl...	200	10.2 KB	3.89 s	
QldXNThLqRwH-OJ1UHjIKG...	200	25.0 KB	3.66 s	
QldKNThLqRwH-OJ1UHjIKG...	200	21.6 KB	3.39 s	

@katie_fenn



Join

Log In

Search packages

Search

Build amazing things

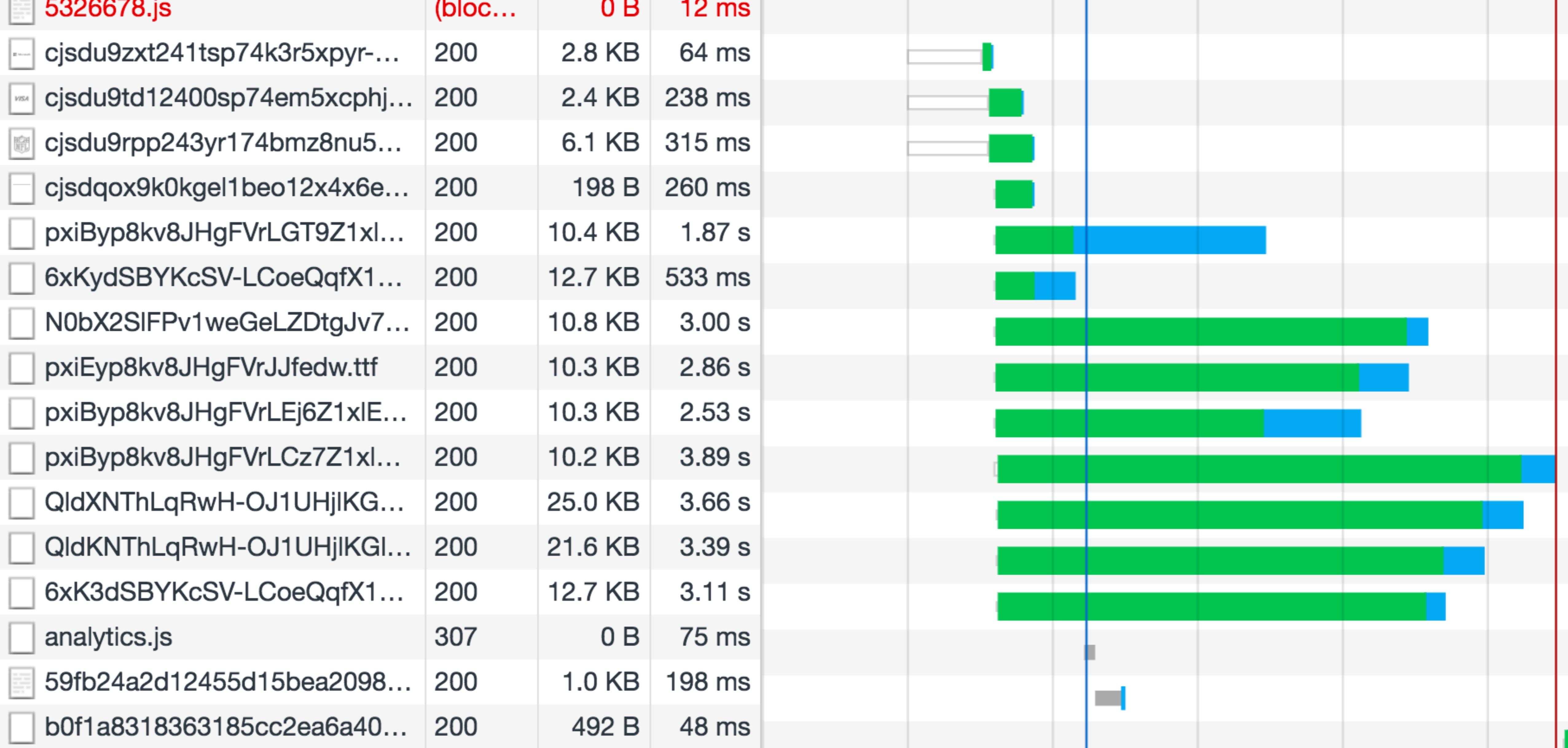
Essential JavaScript development tools that help you go to market faster and build powerful applications using modern open source code.

[See plans](#)

Hit ⌘ R to reload and capture filmstrip.

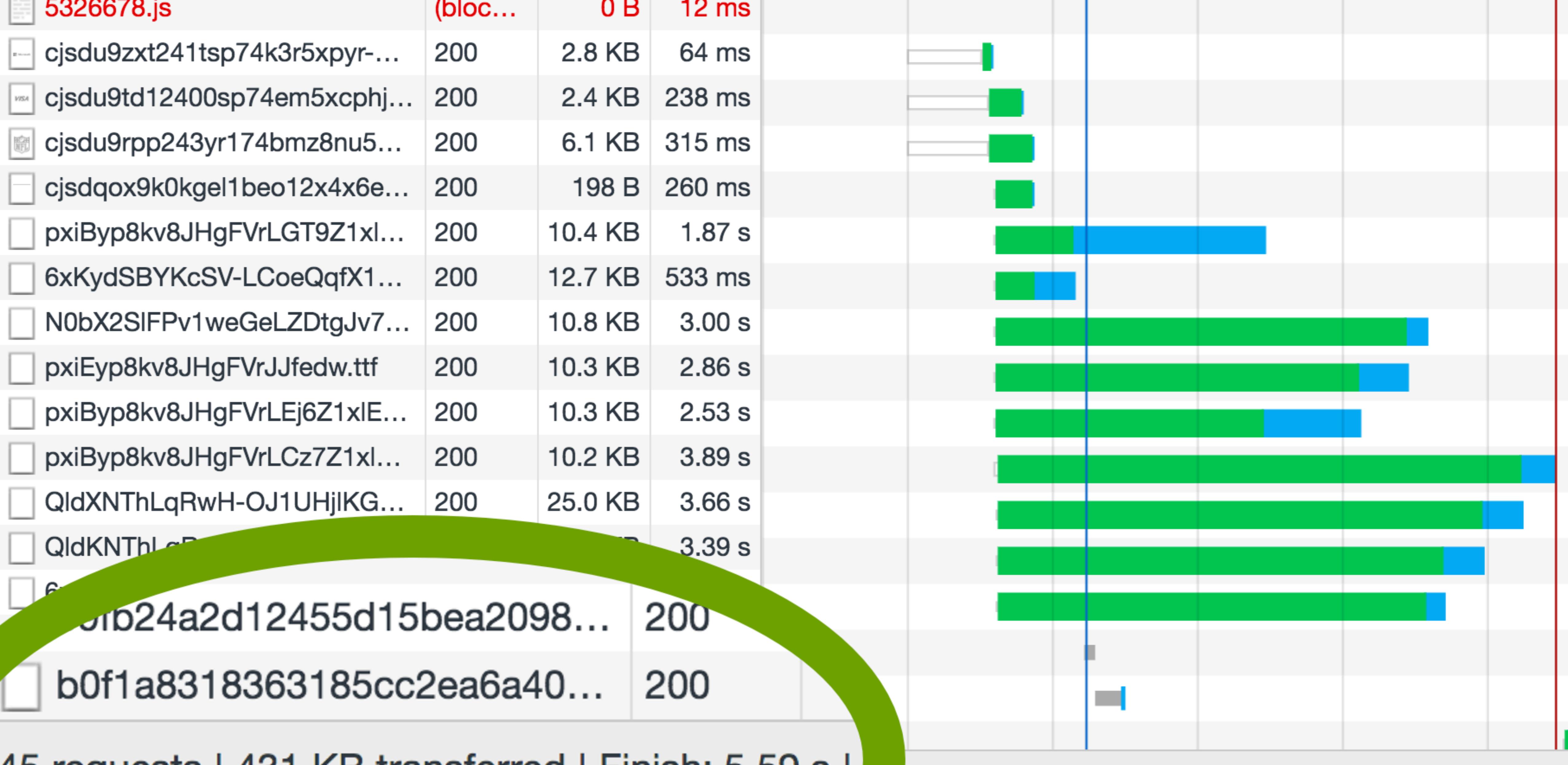
Name	Status	Size	Time	Waterfall
cjsdu9zpy00poph74ndodeid...	200	1.0 KB	200 ms	
cjsdu9zr3241ju674paffp7fz-n...	200	2.3 KB	187 ms	
cjsdu9zk6241qsp74fgw3pn0...	200	3.5 KB	187 ms	
cjsdu9zpy00poph74ndodeid...	200	3.5 KB	364 ms	
5326678.js	(block)	0 B	12 ms	
cjsdu9zxt241tsp74k3r5xpyr...	200	2.8 KB	64 ms	
cjsdu9td12400sp74em5xcphj...	200	2.4 KB	238 ms	
cjsdu9rpp243yr174bmz8nu5...	200	6.1 KB	315 ms	
cjsdqox9k0kgel1beo12x4x6e...	200	198 B	260 ms	
pxiByp8kv8JHgFVrLGT9Z1xl...	200	10.4 KB	1.87 s	
6xKydSBYKcSV-LCoeQqfX1...	200	12.7 KB	533 ms	
N0bX2SIFPv1weGeLZDtgJv7...	200	10.8 KB	3.00 s	
pxiEyp8kv8JHgFVrJJfedw.ttf	200	10.3 KB	2.86 s	
pxiByp8kv8JHgFVrLEj6Z1xE...	200	10.3 KB	2.53 s	
pxiByp8kv8JHgFVrLCz7Z1xl...	200	10.2 KB	3.89 s	
QldXNThLqRwH-OJ1UHjIKG...	200	25.0 KB	3.66 s	
QldKNThLqRwH-OJ1UHjIKG...	200	21.6 KB	3.39 s	

@katie_fenn

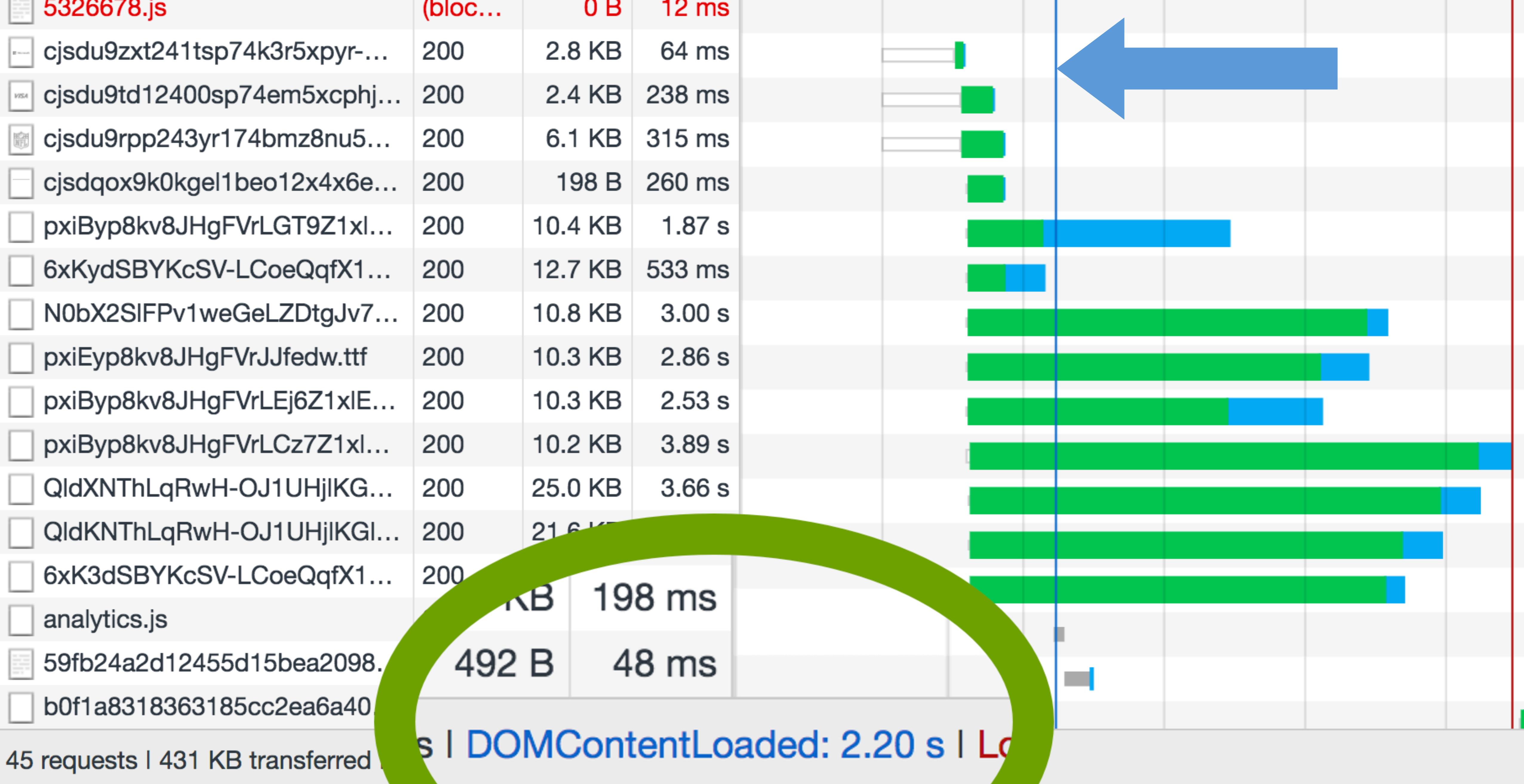


45 requests | 431 KB transferred | Finish: 5.59 s | DOMContentLoaded: 2.20 s | Load: 5.47 s

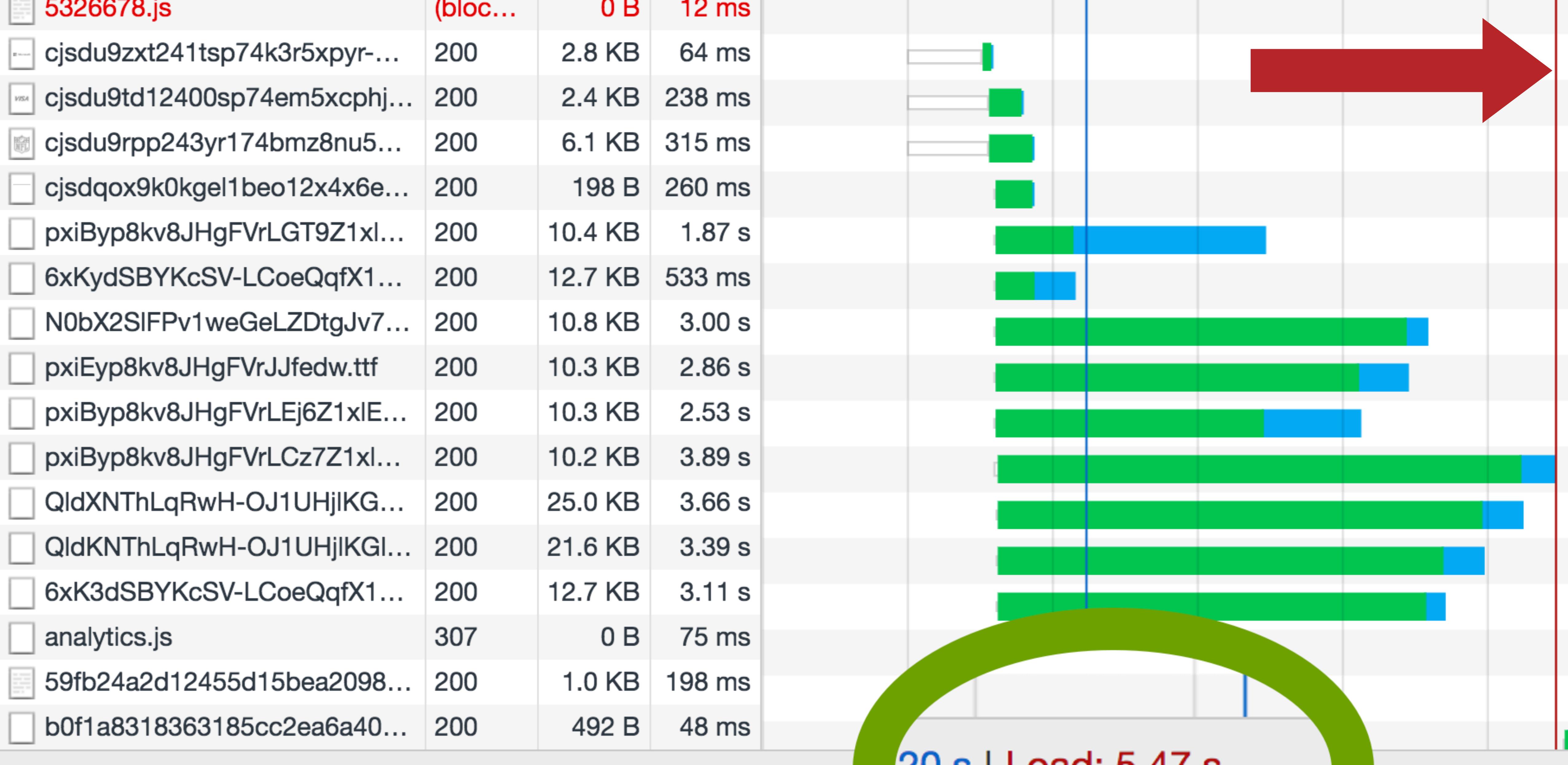
@katie_fenn



@katie_fenn



@katie_fenn



45 requests | 431 KB transferred | Finish: 5.59 s | DOMContentLoaded: 2.

20 s | Load: 5.47 s

@katie_fenn

cw: large animated image

Intermission



Intermission



Performance

Performance

Memory

add stuff to memory



Click the record button  or hit ⌘ E to start a new recording.

Click the reload button  or hit ⌘ ⌂ E to record the page load.

After recording, select an area of interest in the overview by dragging. Then, zoom and pan the timeline with the mousewheel or WASD keys.

[Learn more](#)

Performance

Memory

add stuff to memory



A screenshot of the Chrome DevTools Performance tab. At the top, there are icons for navigation, zoom, and search, followed by tabs for 'Elements', 'Console', 'Sources', and 'Performance'. Below the tabs are buttons for recording ('●'), clearing ('C'), and stopping ('○'). A status message '(no recordings)' is displayed. In the main area, a green circle highlights the recording button icon. The text 'ots Mer' is partially visible on the right.

Click the record button or hit ⌘ E to start a new recording.

Click the reload button or hit ⌘ ⌂ E to record the page load.

After recording, select an area of interest in the overview by dragging. Then, zoom and pan the timeline with the mousewheel or WASD keys.

[Learn more](#)

Performance

Memory

add stuff to memory



A screenshot of the Chrome DevTools Performance tab. At the top, there are icons for navigation, zoom, and search, followed by tabs for 'Elements', 'Console', 'Sources', and 'Performance'. Below the tabs are buttons for recording ('●'), clearing ('C'), and stopping ('○'). A status message '(no recordings)' is displayed. In the main area, a green circle highlights the recording button icon. The text 'ots Mer' is partially visible on the right.

Click the record button or hit ⌘ E to start a new recording.

Click the reload button or hit ⌘ ⌂ E to record the page load.

After recording, select an area of interest in the overview by dragging. Then, zoom and pan the timeline with the mousewheel or WASD keys.

[Learn more](#)

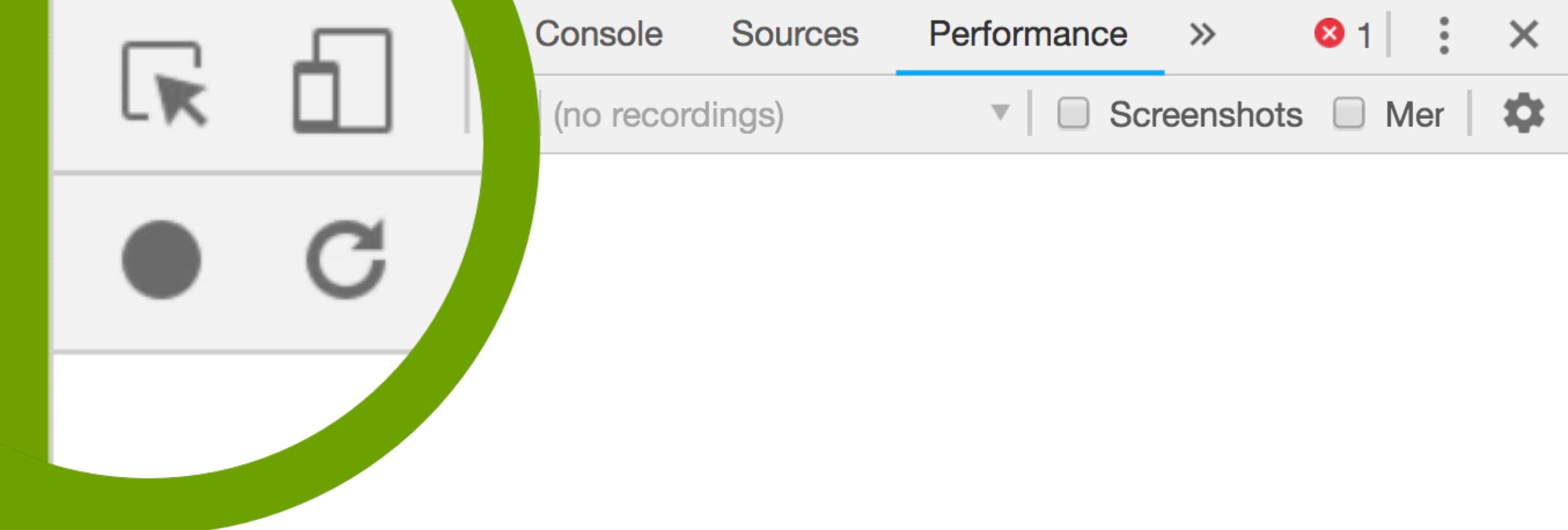
Performance

Memory

add stuff to memory



@katie_fenn



Click the record button or hit ⌘ E to start a new recording.

Click the reload button or hit ⌘ ⌂ E to record the page load.

After recording, select an area of interest in the overview by dragging. Then, zoom and pan the timeline with the mousewheel or WASD keys.

[Learn more](#)

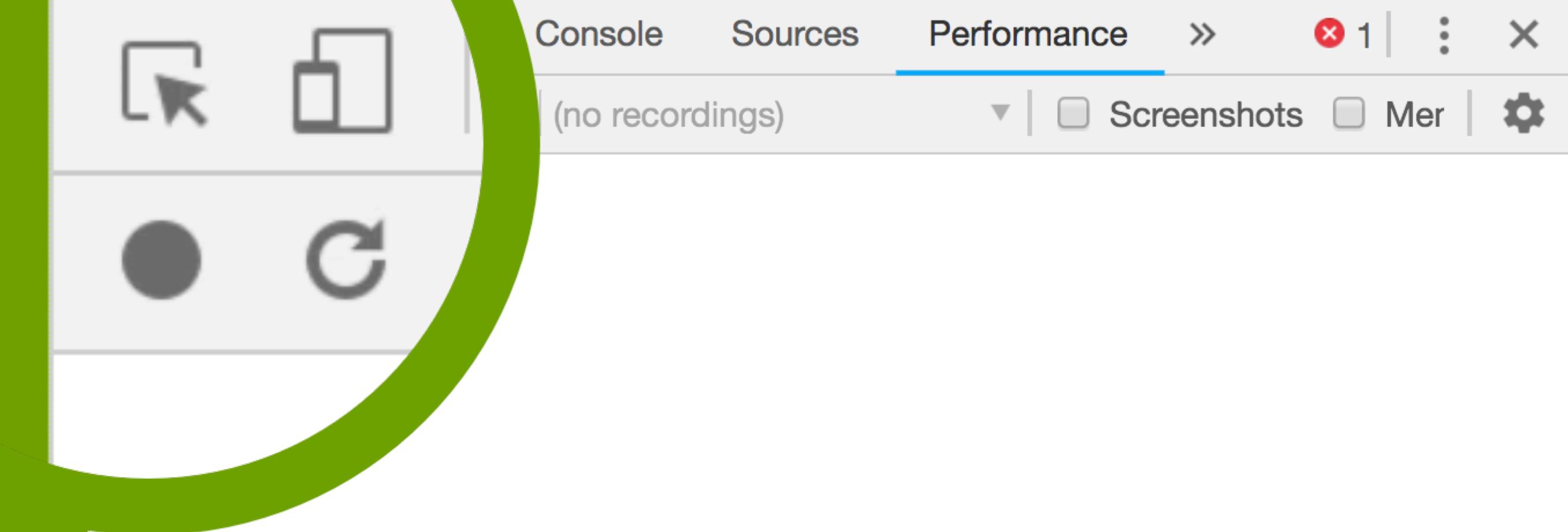
Performance

Memory

add stuff to memory



@katie_fenn



Click the record button or hit ⌘ E to start a new recording.

Click the reload button or hit ⌘ ⌂ E to record the page load.

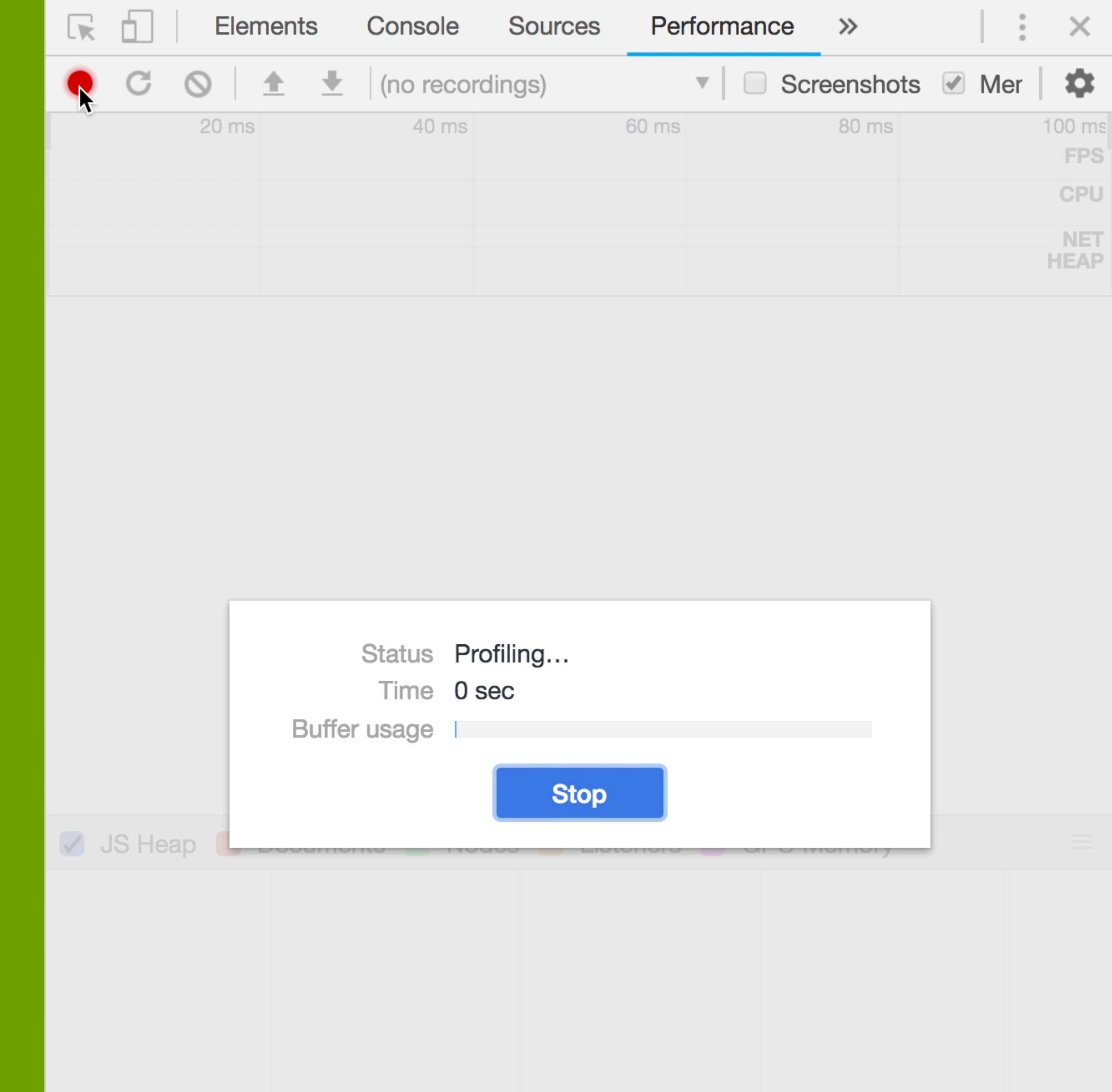
After recording, select an area of interest in the overview by dragging. Then, zoom and pan the timeline with the mousewheel or WASD keys.

[Learn more](#)

Performance

Memory

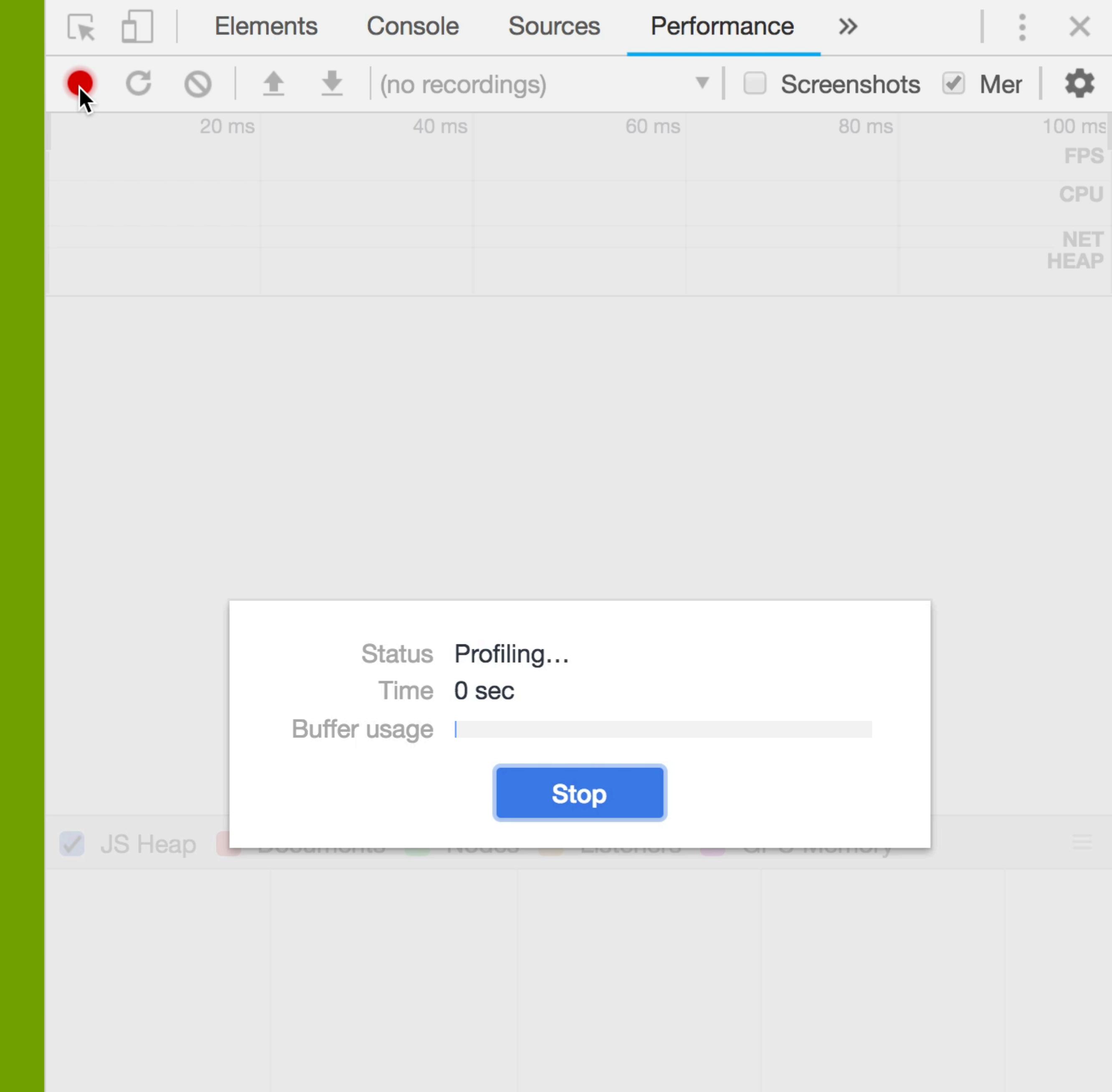
add stuff to memory



Performance

Memory

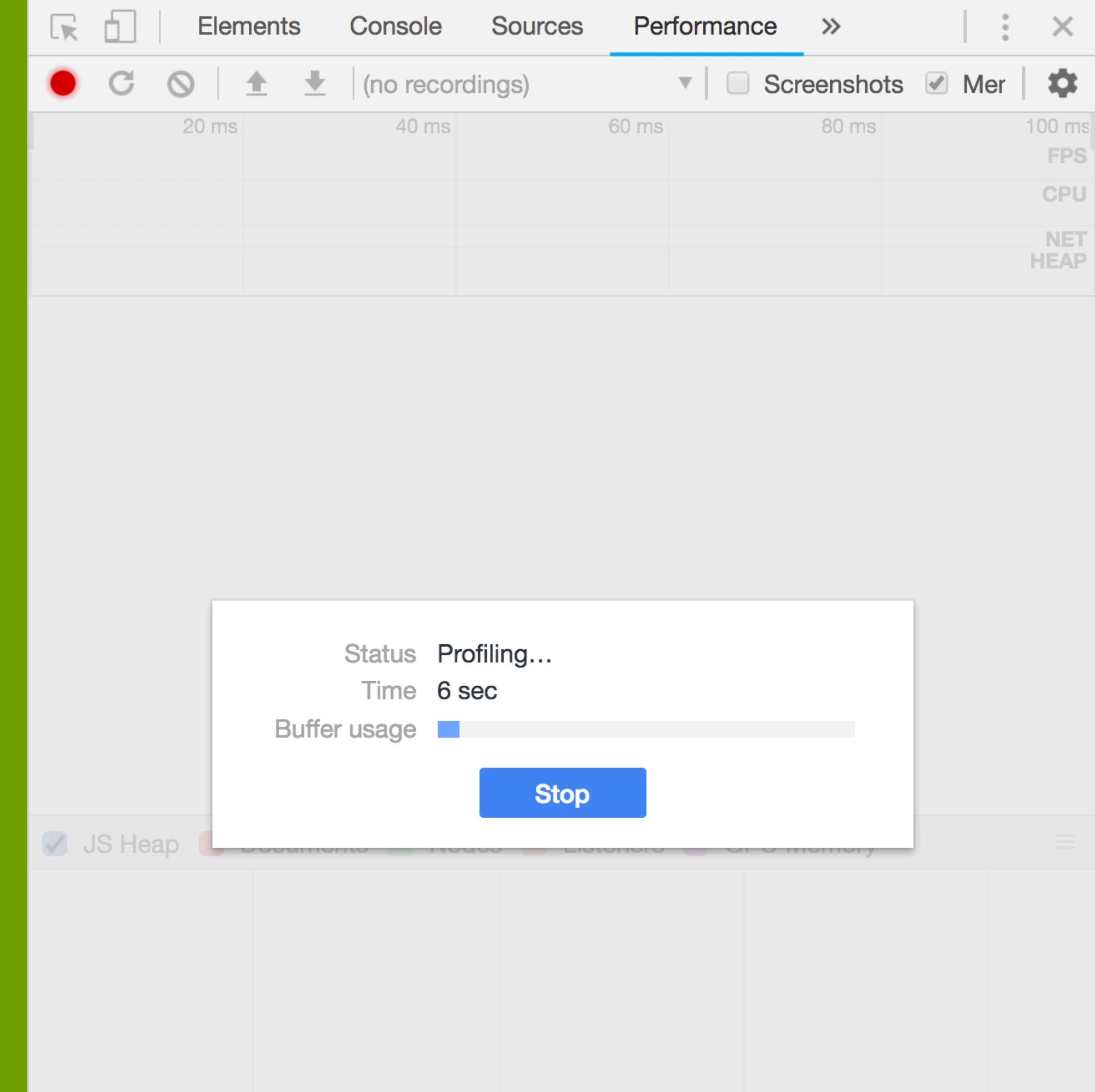
add stuff to memory



Performance

Memory

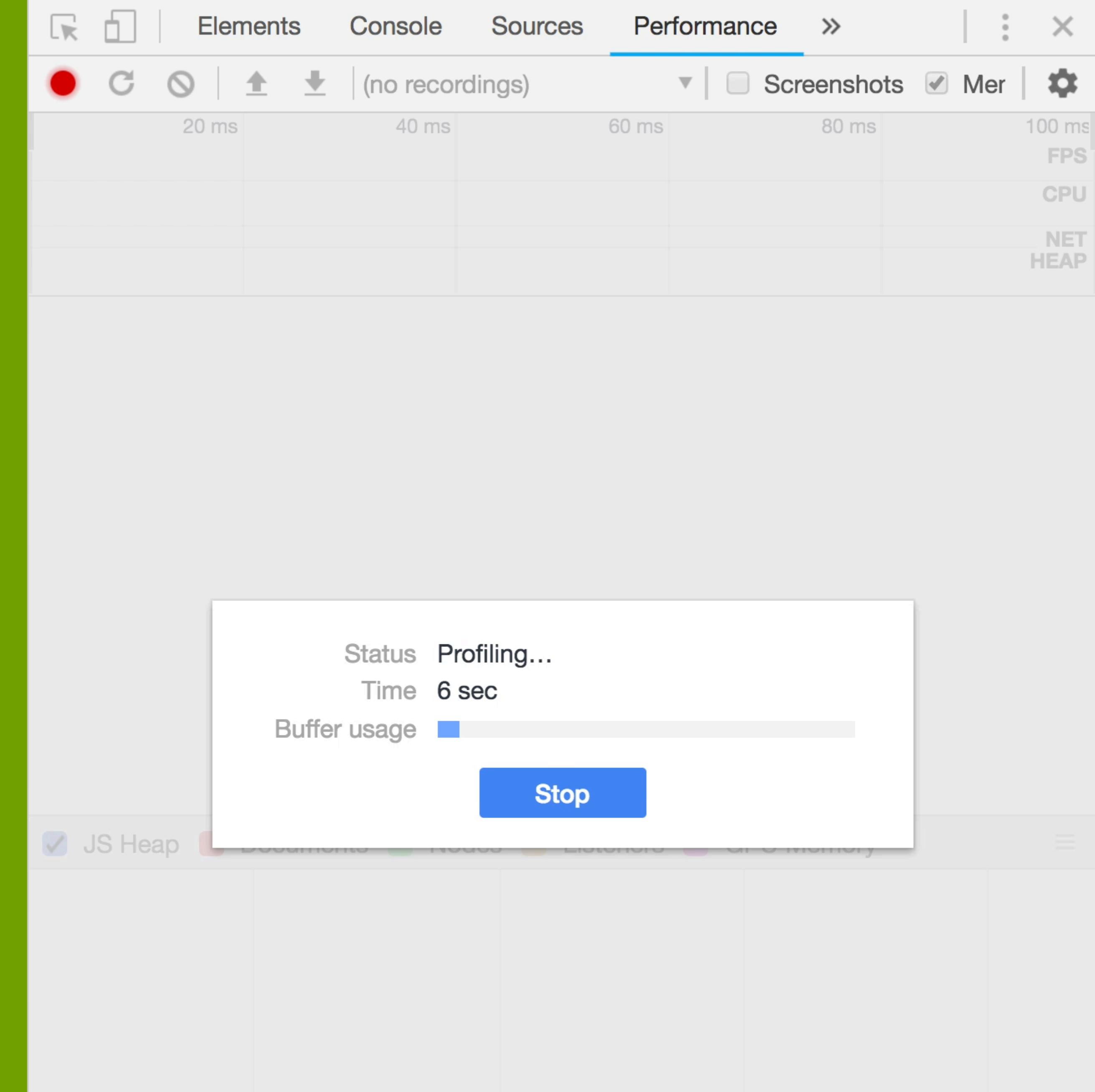
add stuff to memory



Performance

Memory

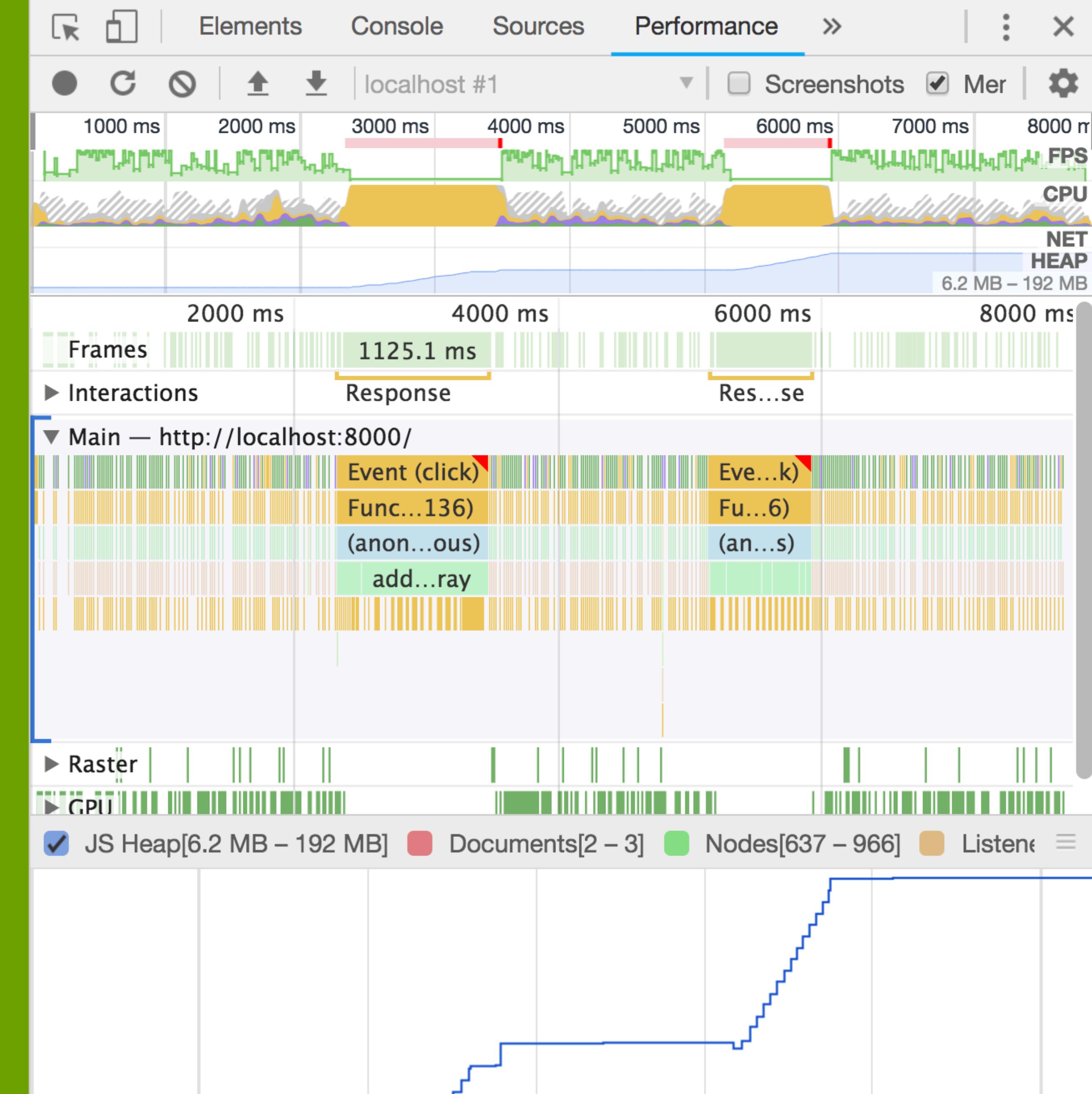
add stuff to memory



Performance

Memory

add stuff to memory

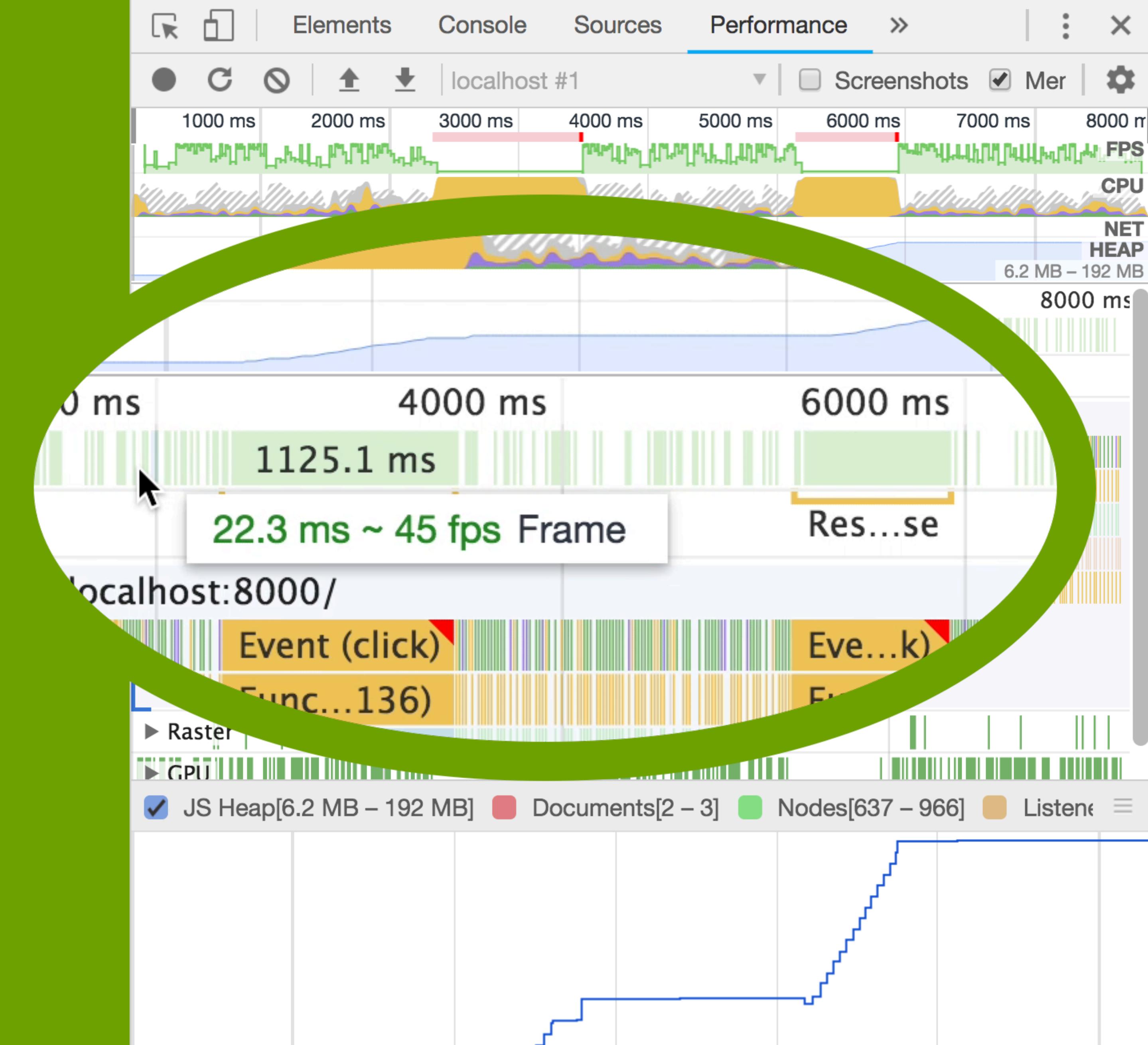


@katie_fenn

Performance

Memory

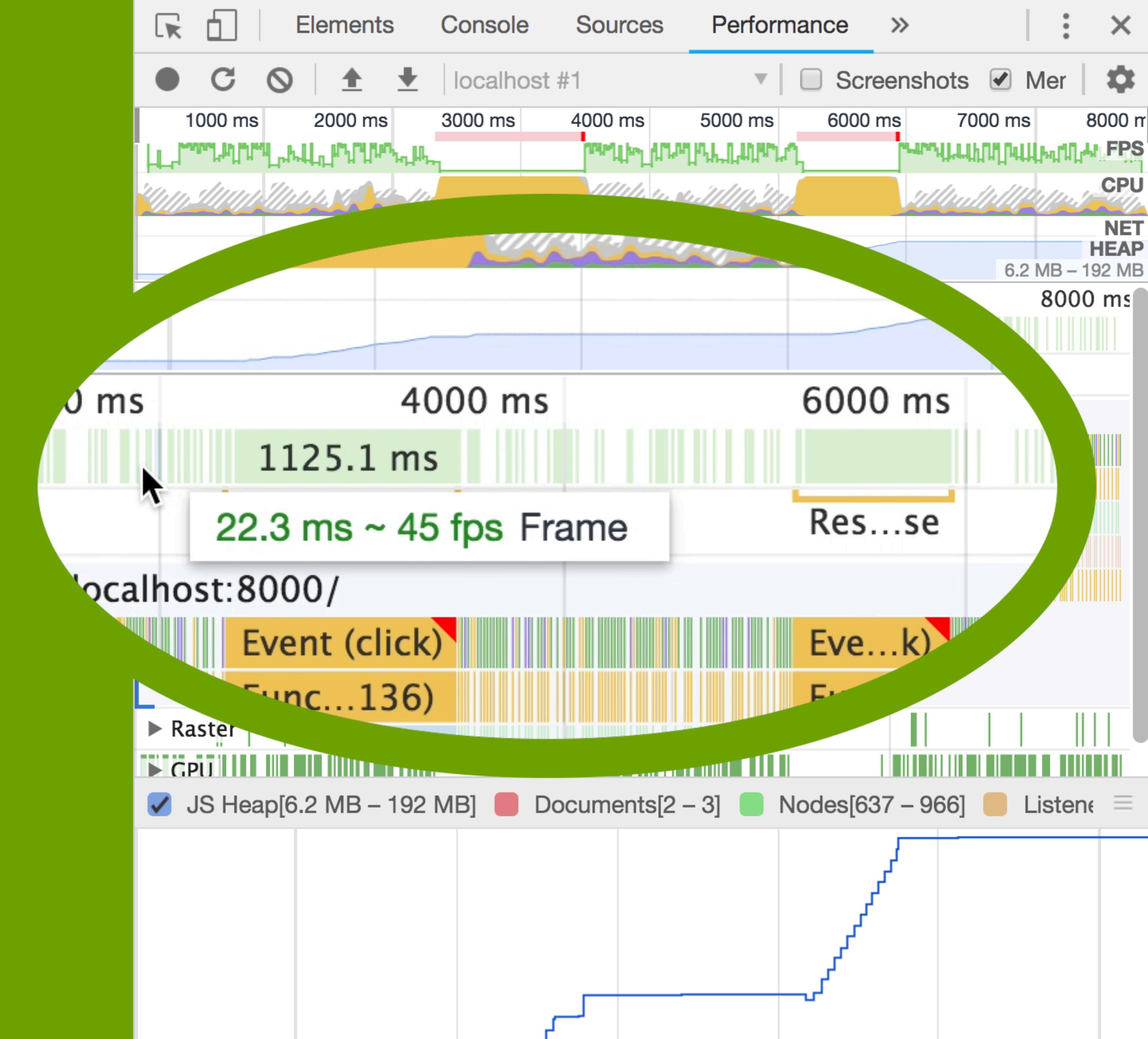
add stuff to memory



Performance

Memory

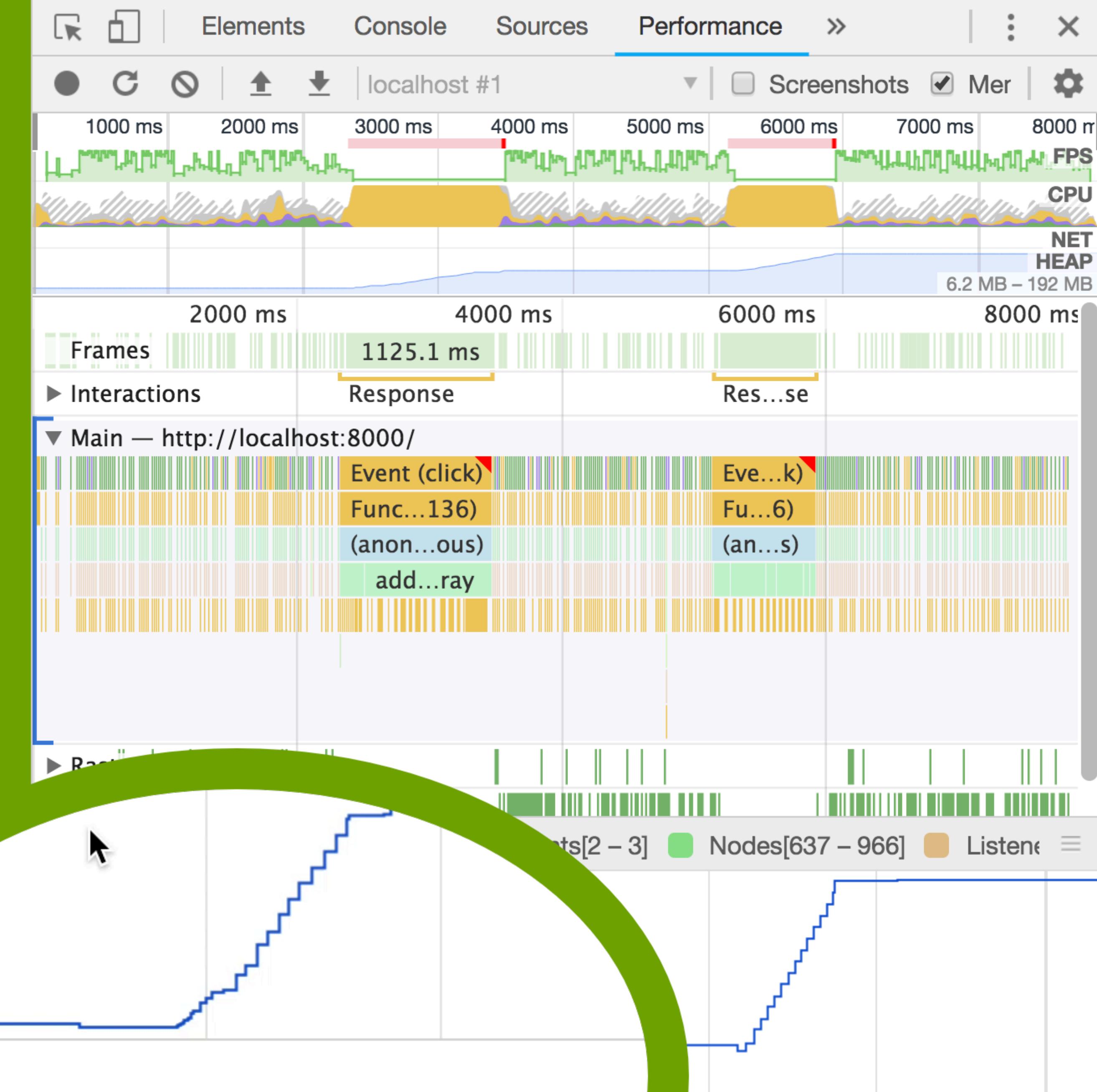
add stuff to memory



Performance

Memory

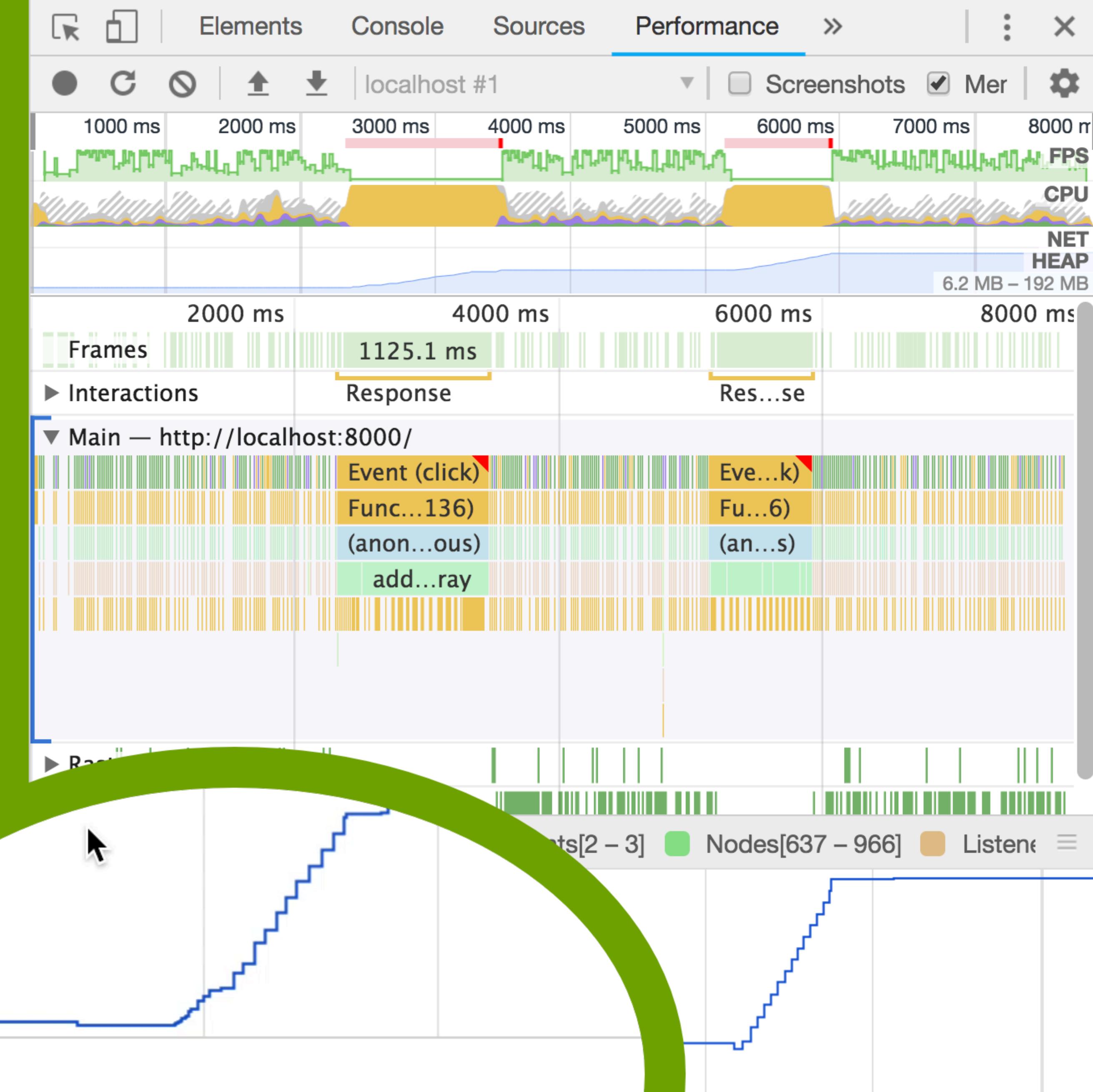
add stuff to memory



Performance

Memory

add stuff to memory



Performance

Memory

add stuff to memory



@katie_fenn

The screenshot shows the Chrome DevTools Performance tab. At the top, there are buttons for recording (a red dot), reloading (a circular arrow), and stopping (a red circle). Below these are options for 'Disable JavaScript samples' and 'Enable advanced paint instrumentation'. A large green circle highlights the main performance timeline area, which is currently blank. In the bottom right corner of the timeline area, a mouse cursor is visible. To the left of the timeline, there are instructions for recording: 'Click the record button' next to a red dot icon, and 'Click the reload button' next to a circular arrow icon. It also says 'or hit ⌘ E' for both actions. Below this, it states 'After recording, select an area of interest in the overview by dragging.' and 'Then, zoom and pan the timeline with the mousewheel or WASD keys.' followed by a link to 'Learn more'.

Click the record button or hit ⌘ E to start a new recording.

Click the reload button or hit ⌘ ⌉ E to record the page load.

After recording, select an area of interest in the overview by dragging.
Then, zoom and pan the timeline with the mousewheel or **WASD** keys.

[Learn more](#)

cw: large animated image

Performance

Memory

add stuff to memory



@katie_fenn

The screenshot shows the Chrome DevTools Performance tab. At the top, there are buttons for recording (a red dot), reloading (a circular arrow), and stopping (a red circle). Below these are options for 'Disable JavaScript samples' and 'Enable advanced paint instrumentation'. A large green circle highlights the main performance timeline area, which is currently blank. In the bottom right corner of the timeline area, a mouse cursor is visible. To the left of the timeline, there are instructions for recording: 'Click the record button' next to a red dot icon, and 'Click the reload button' next to a circular arrow icon. It also says 'or hit ⌘ E' for both actions. Below this, it states 'After recording, select an area of interest in the overview by dragging.' and 'Then, zoom and pan the timeline with the mousewheel or WASD keys.' followed by a link to 'Learn more'.

Click the record button or hit ⌘ E to start a new recording.

Click the reload button or hit ⌘ ⌉ E to record the page load.

After recording, select an area of interest in the overview by dragging.
Then, zoom and pan the timeline with the mousewheel or **WASD** keys.

[Learn more](#)

cw: large animated image

Intermission



Intermission



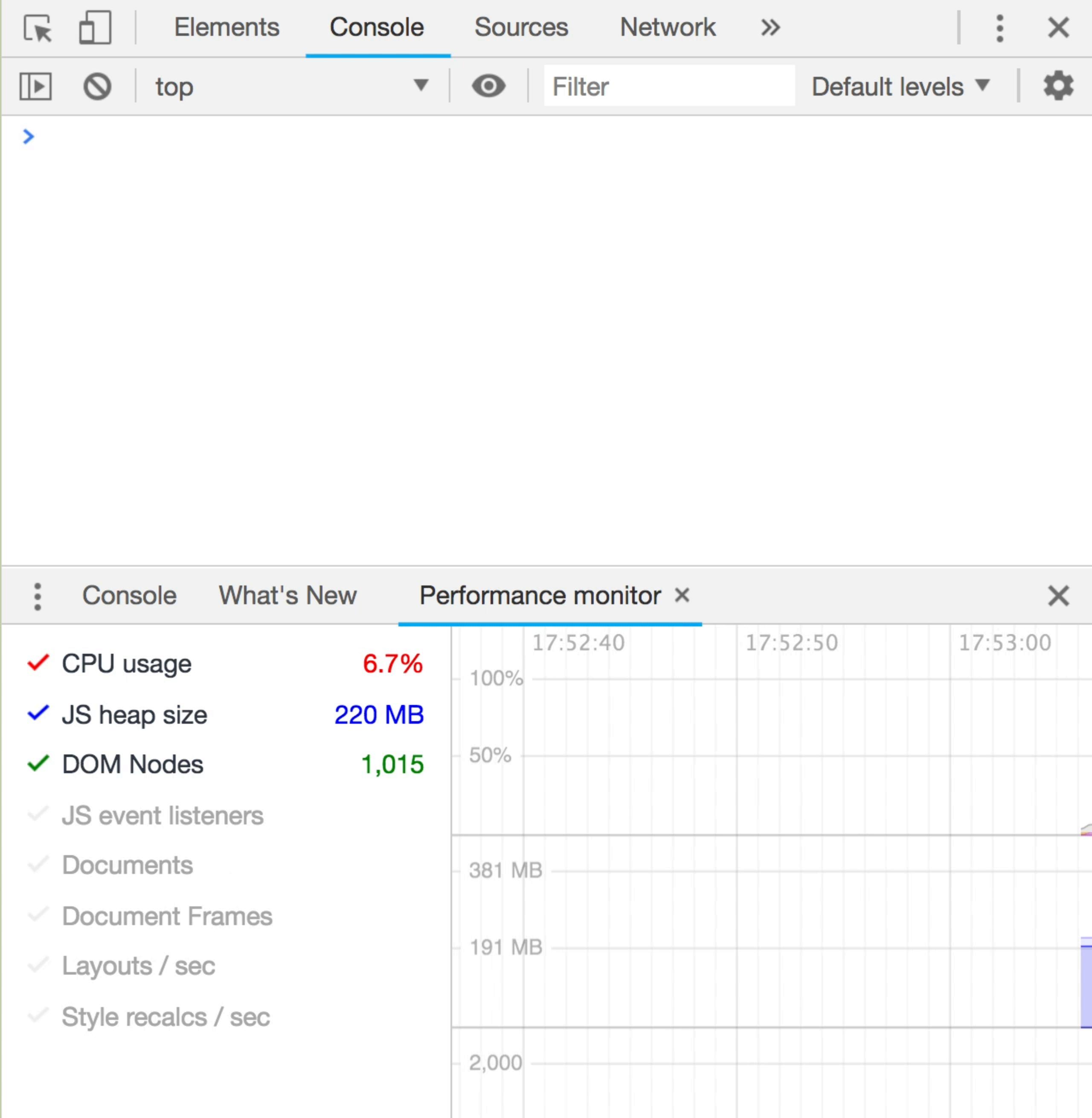
Everything else...

Performance monitor

Performance

Memory

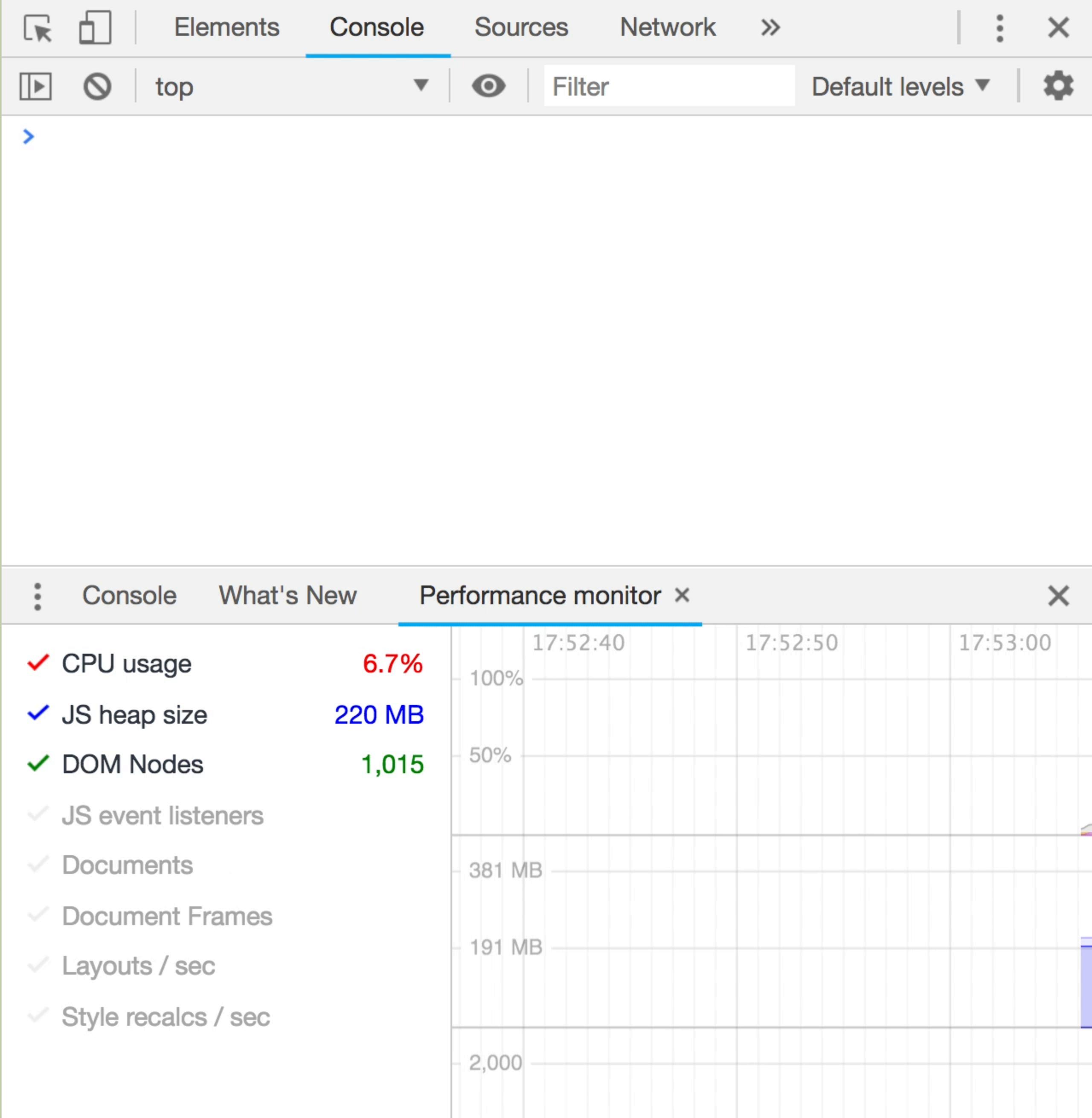
add stuff to memory



Performance

Memory

add stuff to memory



⋮ Console

What's New

Performance monitor ×

X

✓ CPU usage

10.7%

✓ JS heap size

220 MB

✓ DOM Nodes

1,015

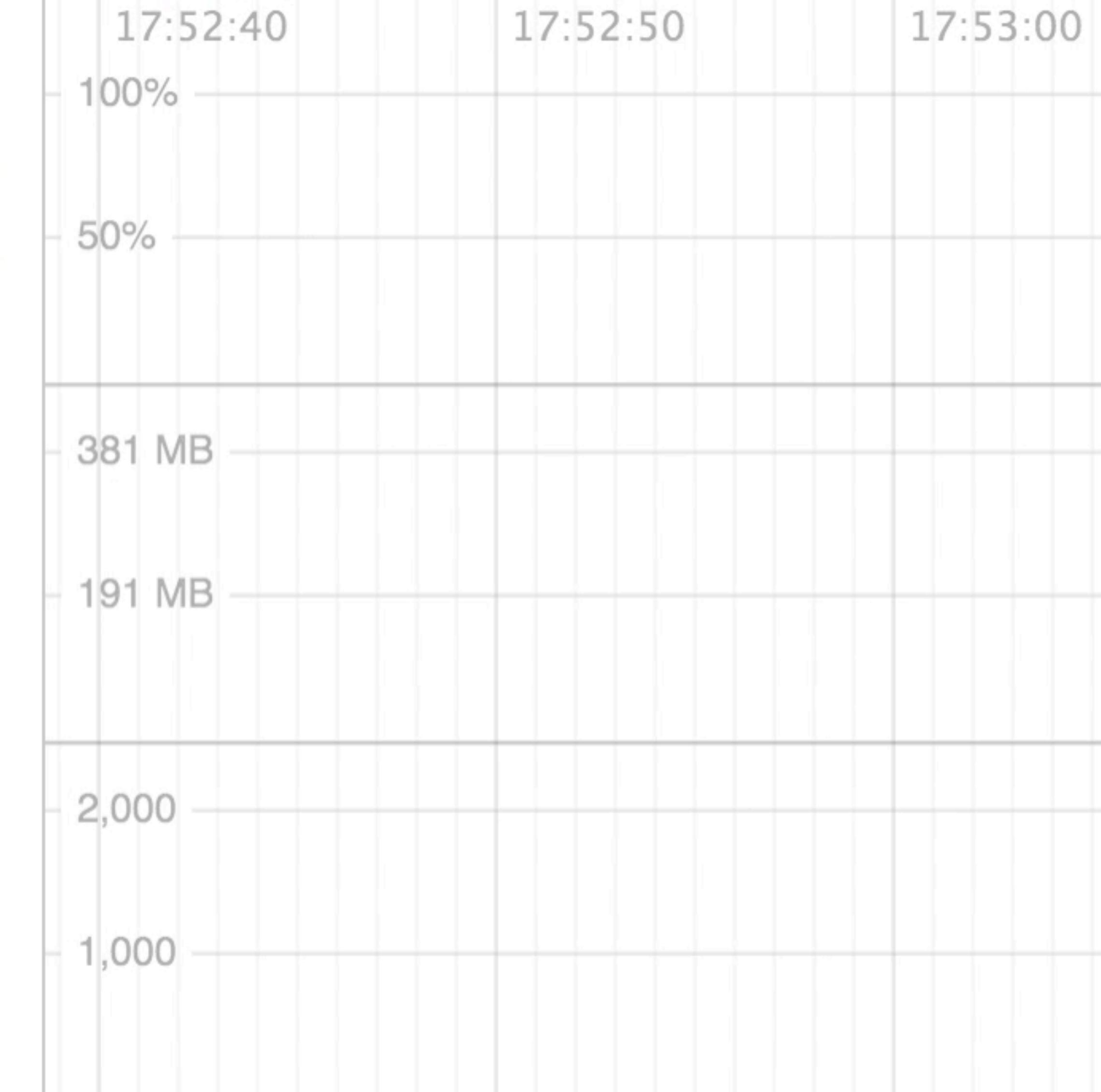
✓ JS event listeners

✓ Documents

✓ Document Frames

✓ Layouts / sec

✓ Style recalcs / sec



⋮ Console

What's New

Performance monitor ×

X

✓ CPU usage

10.7%

✓ JS heap size

220 MB

✓ DOM Nodes

1,015

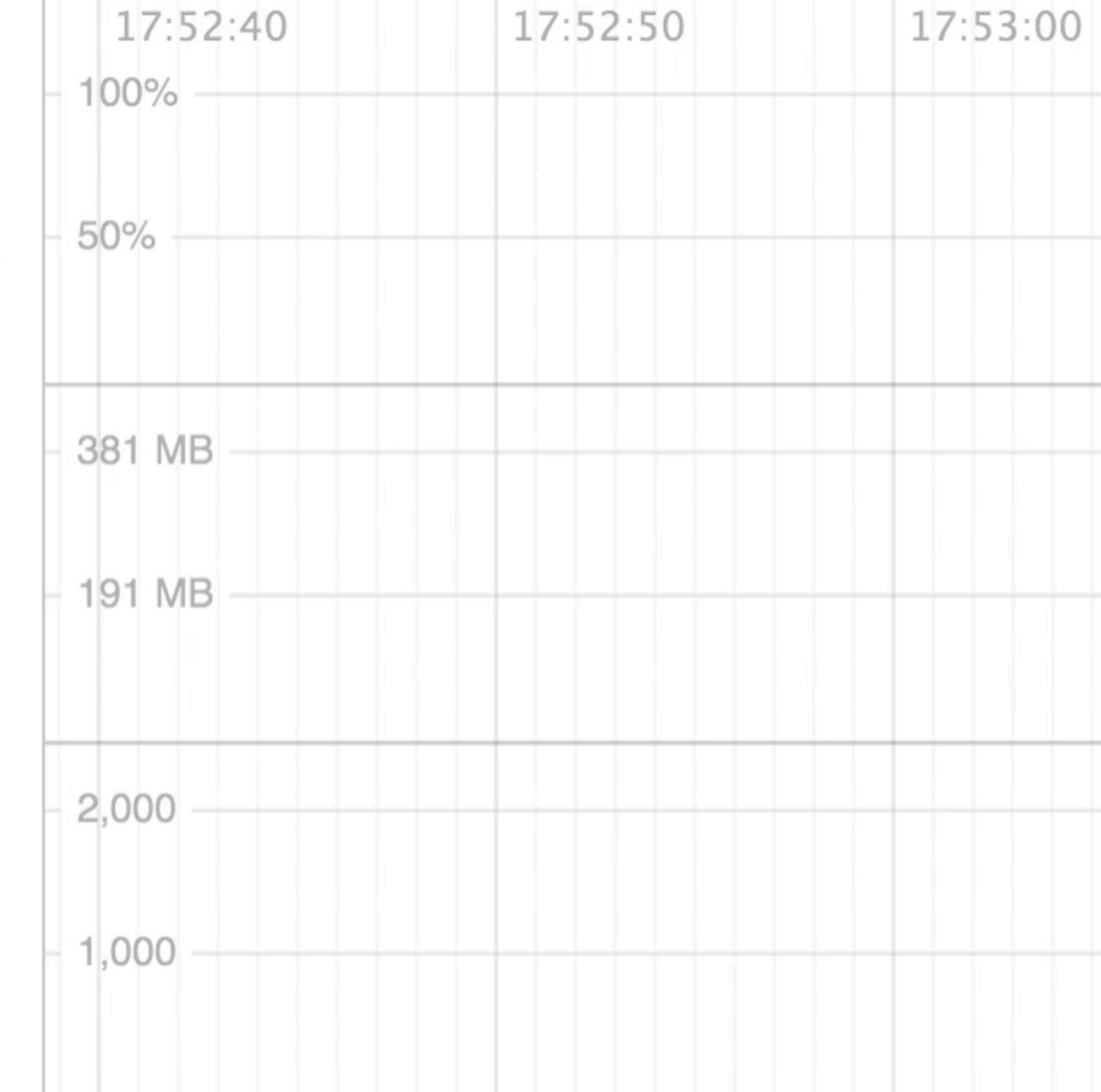
✓ JS event listeners

✓ Documents

✓ Document Frames

✓ Layouts / sec

✓ Style recalcs / sec



Command menu

Command menu

Shift + Cmd / Ctrl + P



Command menu

Shift + Cmd / Ctrl + P



Dark mode



Dark mode

@katie_fenn



Dark mode

@katie_fenn

CPU profiling

457px × 800px

JavaScript profiling

call long function

call short function



Record JavaScript CPU Profile

Record JavaScript CPU Profile

CPU profiles show where the execution time is spent in your page's JavaScript functions.

Select JavaScript VM instance

7.0 MB / 11.0 MB localhost:8000

Start

Load

457px × 800px

JavaScript profiling

call long function

call short function



Record JavaScript CPU Profile

Record JavaScript CPU Profile

CPU profiles show where the execution time is spent in your page's JavaScript functions.

Select JavaScript VM instance

7.0 MB / 11.0 MB localhost:8000

Start

Load

457px × 800px

JavaScript profiling

call long function

call short function



Record JavaScript CPU Profile

Record JavaScript CPU Profile

CPU profiles show where the execution time is spent in your page's JavaScript functions.

Select JavaScript VM instance

7.0 MB / 11.0 MB localhost:8000

Start

Load

457px × 800px

JavaScript profiling

call long function

call short function



Record JavaScript CPU Profile

Record JavaScript CPU Profile

CPU profiles show where the execution time is spent in your page's JavaScript functions.

Select JavaScript VM instance

7.0 MB / 11.0 MB localhost:8000

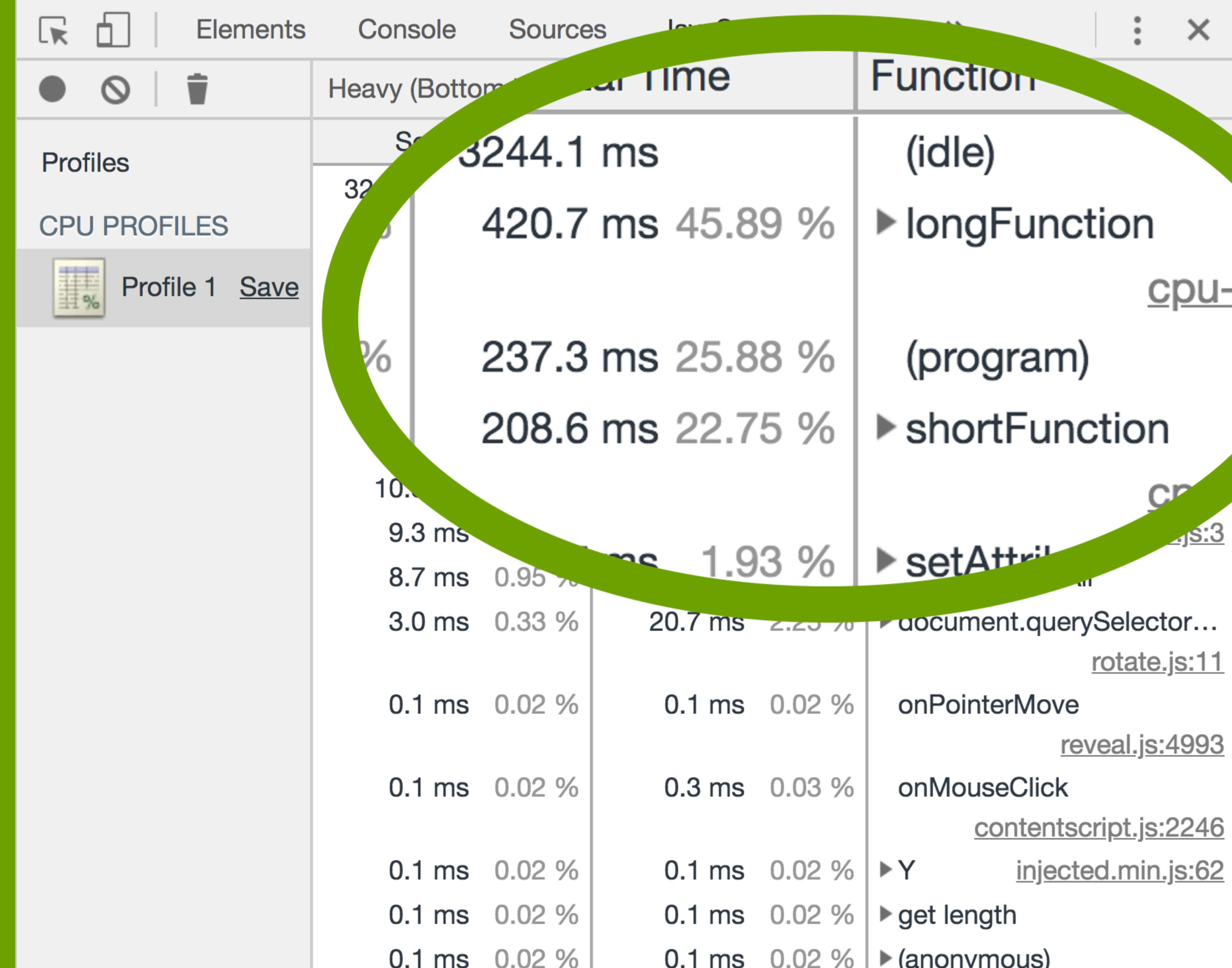
Start

Load

JavaScript profiling

call long function

call short function



JavaScript profiling

call long function

call short function

Profiles	Self Time	Total Time	Function
CPU PROFILES	3244.1 ms	3244.1 ms	(idle)
Profile 1 Save	420.7 ms 45.89 %	420.7 ms 45.89 %	► longFunction cpu-loader.js:1
	237.3 ms 25.88 %	237.3 ms 25.88 %	(program)
	208.6 ms 22.75 %	208.6 ms 22.75 %	► shortFunction cpu-loader.js:7
	17.7 ms 1.93 %	17.7 ms 1.93 %	► setAttribute
	10.8 ms 1.18 %	10.8 ms 1.18 %	► requestAnimationFrame
	9.3 ms 1.01 %	49.6 ms 5.40 %	rotate rotate.js:3
	8.7 ms 0.95 %	8.7 ms 0.95 %	► querySelectorAll
	3.0 ms 0.33 %	20.7 ms 2.25 %	► document.querySelector... rotate.js:11
	0.1 ms 0.02 %	0.1 ms 0.02 %	onPointerMove reveal.js:4993
	0.1 ms 0.02 %	0.3 ms 0.03 %	onMouseClick contentscript.js:2246
	0.1 ms 0.02 %	0.1 ms 0.02 %	► Y injected.min.js:62
	0.1 ms 0.02 %	0.1 ms 0.02 %	► get length
	0.1 ms 0.02 %	0.1 ms 0.02 %	► (anonymous)

@katie_fenn

JavaScript profiling

call long function

call short function

Profiles	Self Time	Total Time	Function
CPU PROFILES	3244.1 ms	3244.1 ms	(idle)
Profile 1 Save	420.7 ms 45.89 %	420.7 ms 45.89 %	► longFunction cpu-loader.js:1
	237.3 ms 25.88 %	237.3 ms 25.88 %	(program)
	208.6 ms 22.75 %	208.6 ms 22.75 %	► shortFunction cpu-loader.js:7
	17.7 ms 1.93 %	17.7 ms 1.93 %	► setAttribute
	10.8 ms 1.18 %	10.8 ms 1.18 %	► requestAnimationFrame
	9.3 ms 1.01 %	49.6 ms 5.40 %	rotate rotate.js:3
	8.7 ms 0.95 %	8.7 ms 0.95 %	► querySelectorAll
	3.0 ms 0.33 %	20.7 ms 2.25 %	► document.querySelector... rotate.js:11
	0.1 ms 0.02 %	0.1 ms 0.02 %	onPointerMove reveal.js:4993
	0.1 ms 0.02 %	0.3 ms 0.03 %	onMouseClick contentscript.js:2246
	0.1 ms 0.02 %	0.1 ms 0.02 %	► Y injected.min.js:62
	0.1 ms 0.02 %	0.1 ms 0.02 %	► get length
	0.1 ms 0.02 %	0.1 ms 0.02 %	► (anonymous)

@katie_fenn

Node.JS debugging

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help

2. kas@kiki: ~/Documents/Talks/Chrome Devtools 2019 (zsh)

18:38:28 >

kas@kiki ~ ~/Documents/Talks/Chrome Devtools 2019

18:38:31 > |

@katie_fenn

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help

2. kas@kiki: ~/Documents/Talks/Chrome Devtools 2019 (zsh)

18:38:28 >

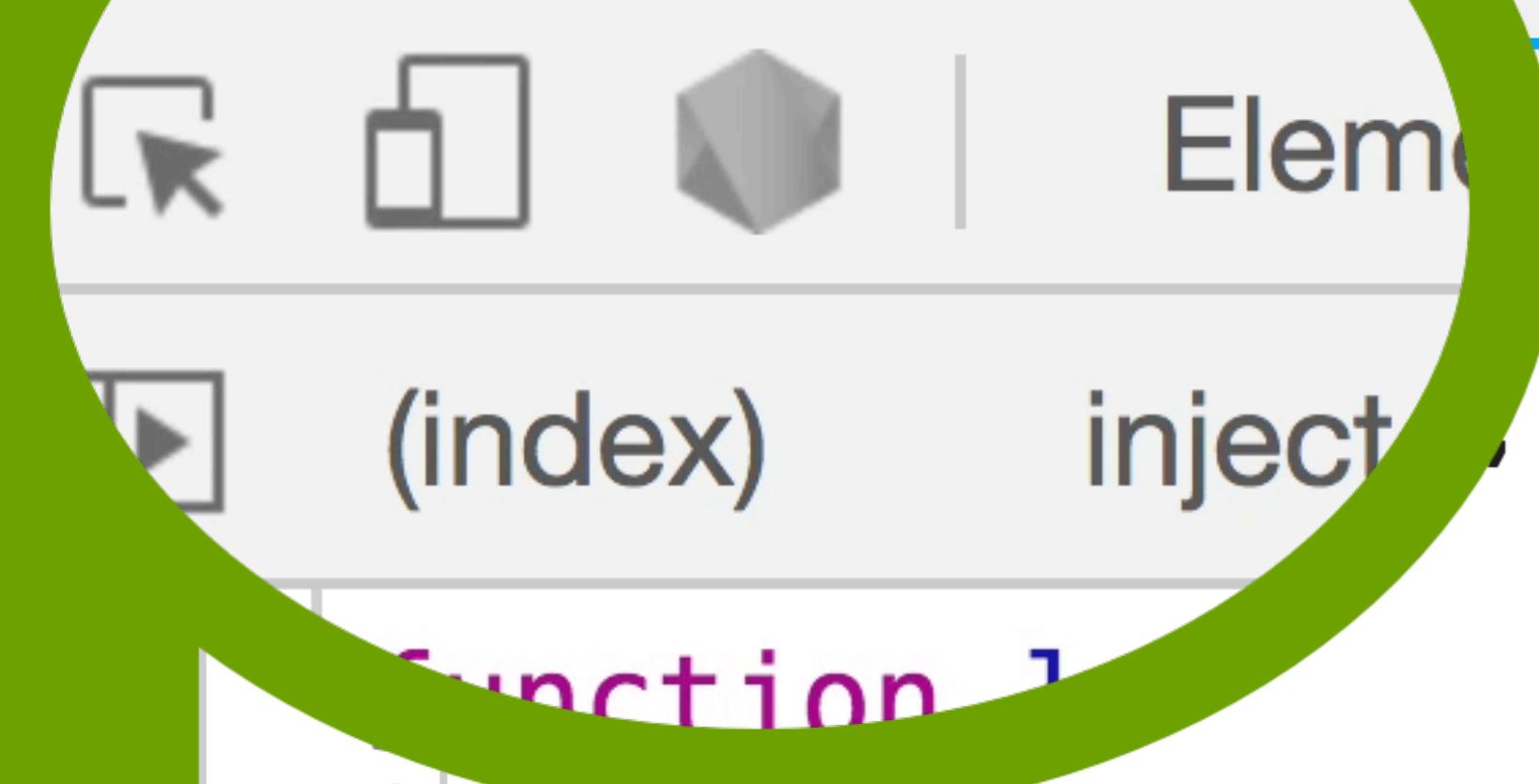
kas@kiki ~ ~/Documents/Talks/Chrome Devtools 2019

18:38:31 > |

@katie_fenn

Node.JS debugging

node --inspect-brk



The screenshot shows the Node.js debugger interface. At the top, there are icons for back, forward, and element inspection. The tab bar shows "cpu-loader.js" is active. Below the tabs, the code editor displays a function definition:

```
function () {
  inject()
}

function shortFunction () {
  for (var index = 0; index < 500000000; index++) {
}
```

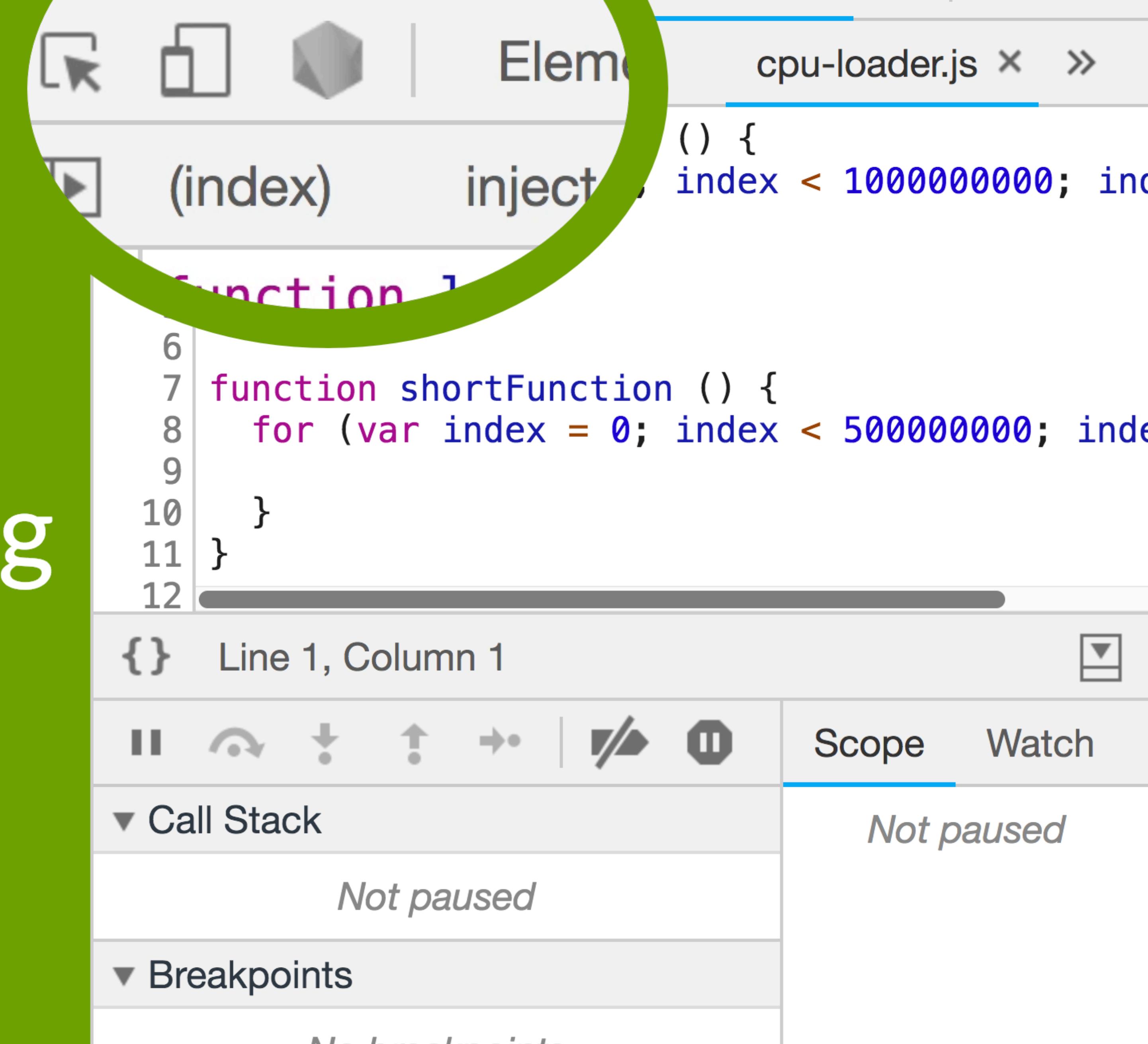
The line 12 of the code is highlighted with a red bar. The status bar below the code editor says "Line 1, Column 1".

Below the code editor is a toolbar with various debugging icons: pause, resume, step over, step into, step out, and a refresh symbol. To the right of the toolbar are two tabs: "Scope" and "Watch", with "Scope" being the active tab. The status message "Not paused" is displayed next to the tabs.

Below the toolbar are three expandable sections: "Call Stack", "Breakpoints", and "Scope". The "Call Stack" section is expanded, showing "Not paused". The "Breakpoints" section is also expanded, showing "No breakpoints".

Node.JS debugging

node --inspect-brk



The screenshot shows the Node.js debugger interface. At the top, there are icons for back, forward, and element inspection. The tab bar shows "cpu-loader.js" is active. Below the tabs, the code editor displays a function definition:

```
function () {
  inject()
}

function shortFunction () {
  for (var index = 0; index < 500000000; index++) {
}
```

The line 12 of the code is highlighted with a green background. The status bar at the bottom indicates "Not paused".

Below the code editor, the "Scope" tab is selected, showing "Line 1, Column 1". The "Call Stack" and "Breakpoints" sections are also visible.

Node.JS debugging

node --inspect-brk

The screenshot shows the Chrome DevTools interface with the 'Sources' tab selected. A tooltip is displayed over the green hexagonal icon, reading 'Open dedicated DevTools for Node.js'. The code editor displays two functions:

```
1 function longFunction () {  
2     for (var index = 0; index < 1000000000; index++) {  
3         //...  
4     }  
5 }  
6  
7 function shortFunction () {  
8     for (var index = 0; index < 500000000; index++) {  
9         //...  
10    }  
11 }  
12
```

The status bar at the bottom indicates 'Line 1, Column 1'. The debugger controls include a pause button, step buttons, and dropdowns for 'Scope' and 'Watch'. The 'Call Stack' and 'Breakpoints' sections are shown below, both labeled 'Not paused'.

Node.JS debugging

node --inspect-brk

The screenshot shows the Chrome DevTools interface with the 'Sources' tab selected. A tooltip is displayed over the green hexagonal icon in the top bar, reading 'Open dedicated DevTools for Node.js'. The main area displays two functions:

```
1 function longFunction () {  
2     for (var index = 0; index < 1000000000; index++) {  
3         //...  
4     }  
5 }  
6  
7 function shortFunction () {  
8     for (var index = 0; index < 500000000; index++) {  
9         //...  
10    }  
11 }  
12
```

The code editor highlights the first line of the 'longFunction' block. Below the code, the status bar shows 'Line 1, Column 1'. The bottom section of the DevTools includes a toolbar with icons for step, refresh, and search, followed by tabs for 'Scope' and 'Watch', both of which are currently inactive ('Not paused'). Three sections are listed: 'Call Stack', 'Breakpoints', and 'Breakpoints' (repeated), all showing 'Not paused'.

Lighthouse

The audits panel

Katie Fenn retrospective

28th July 2018



It was an emotional moment when [Peter Aitken](#) announced that the 2018 edition of <http://scotlandjs.com/> was to be the last. This is not a normal reaction to the end of a tech conference, but Scotland JS is not just a normal event.

I became involved with Scotland JS in early 2015. I'd delivered my first conference talk during the autumn of 2014, and was wondering where to go next. Peter



Audits

Identify and fix common problems that affect your site's performance, accessibility, and user experience. [Learn more](#)

Device

- Mobile
- Desktop

Audits

- Performance
- Progressive Web App
- Best practices
- Accessibility
- SEO

Throttling

- Simulated Fast 3G, 4x CPU Slowdown
- Applied Fast 3G, 4x CPU Slowdown
- No throttling

- Clear storage

[Run audits](#)

Katie Fenn retrospective

28th July 2018



It was an emotional moment when [Peter Aitken](#) announced that the 2018 edition of <http://scotlandjs.com/> was to be the last. This is not a normal reaction to the end of a tech conference, but Scotland JS is not just a normal event.

I became involved with Scotland JS in early 2015. I'd delivered my first conference talk during the autumn of 2014, and was wondering where to go next. Peter



Audits

Identify and fix common problems that affect your site's performance, accessibility, and user experience. [Learn more](#)

Device

- Mobile
- Desktop

Audits

- Performance
- Progressive Web App
- Best practices
- Accessibility
- SEO

Throttling

- Simulated Fast 3G, 4x CPU Slowdown
- Applied Fast 3G, 4x CPU Slowdown
- No throttling

- Clear storage

[Run audits](#)

But what about Firefox?

Firefox Developer Tools

Style editing

Class toggling

Colour pickers

JS console

`console.log`

`console.dir`

`console.time`

`console.trace`

Step-through debugging

Breakpoints

Pause on exception

Network profiling

Network throttling

Animation profiling

Memory profiling

Dark theme

CPU profiling

Performance monitor

Accessibility audits

Let's **work** the problem.
Let's not make things
worse **by** guessing.



cw: large animated image



You guys, this is so
f***ing cool.

Thank you

twitter:

@katie_fenn



You guys, this is so
f***ing cool.

Thank you

twitter:

@katie_fenn