

# PRAGUEMATIC

## WEB PERFORMANCE

**DENYS MISHUNOV**

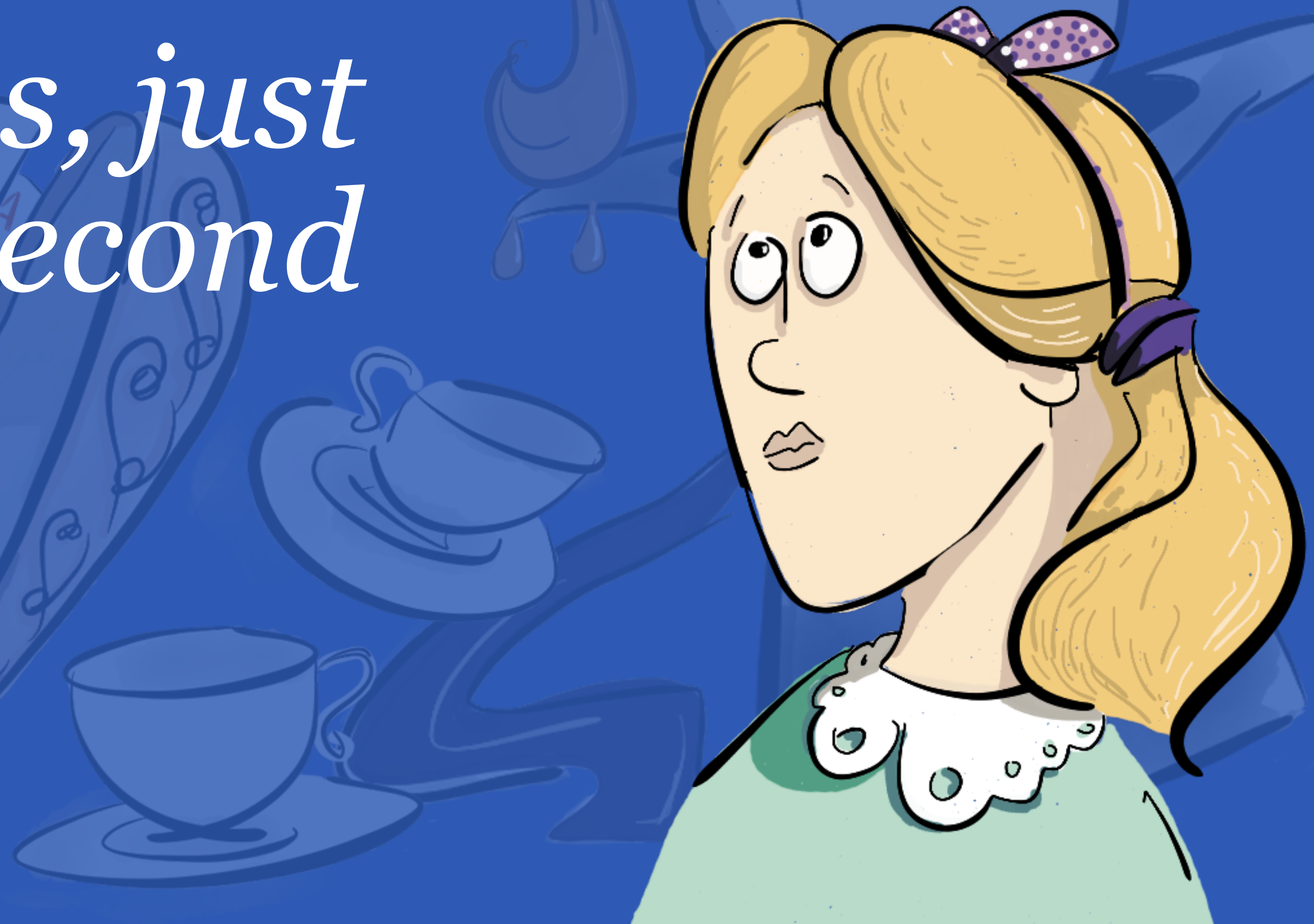
Senior Frontend Engineer, Create::Editor



IT'S ABOUT  
**FRONT-END**

*How long is  
forever?*

*Sometimes, just  
one second*





blah-blah  
BLAH-BLAH  
blah-blah

# PRAGUEMATIC WEB PERFORMANCE

# WEB PERFORMANCE 2020



***“OFTEN POOR  
RESPONSE  
TIME ON  
WEBSITE”***



***“PERFORMANCE...  
OTHERWISE A  
GREAT TOOL”***



***“IT'S JUST SLOW”***





“OFTEN POOR RESPONSE  
TIME ON WEBSITE”

“IT'S JUST SLOW”

“PERFORMANCE...  
OTHERWISE A GREAT  
TOOL”

“OFTEN POOR RESPONSE  
TIME ON WEBSITE”

# 18.5%

“PERFORMANCE...  
OTHERWISE A GREAT

Dedicated the time to say how *not impressed* they are with performance

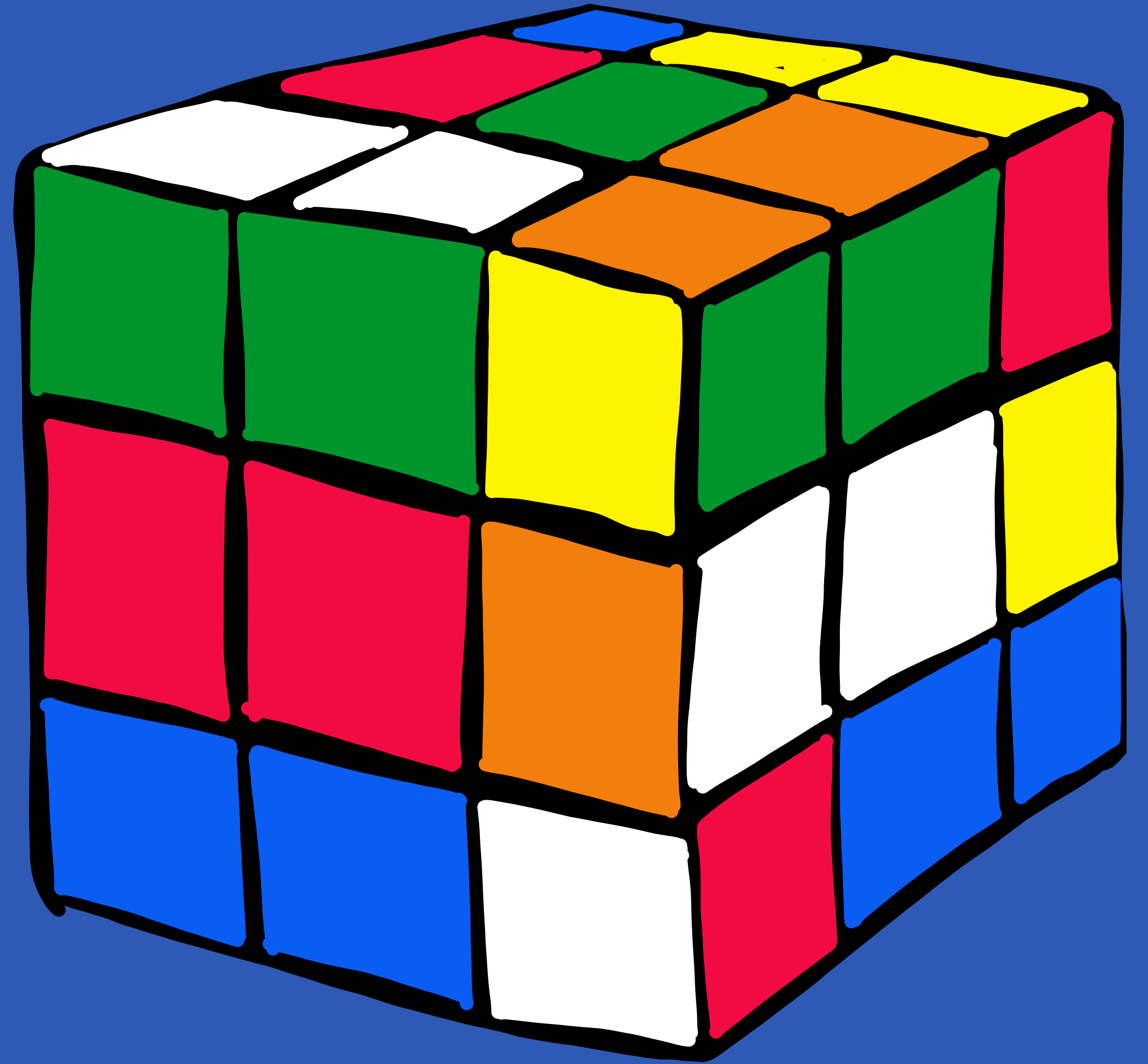
“IT'S JUST SLOW”

GitLab, UX research by **Jeff Crow** and **Farnoosh Seifoddini**

# Q2 OKRs

<https://gitlab.com/groups/gitlab-com/-/epics/464>

# **WEB PERFORMANCE 2020**



# PRAGMATIC

prag • mat • ic

adj. Dealing or concerned with facts or actual occurrences; practical

***“IT'S OK, BUT SPEED MAY  
BE IMPROVED”***

**MONITOR • OPTIMISE**

**MEASURE • MONITOR • OPTIMISE**

# **PRAGMATIC**

**MEASURE • MONITOR • OPTIMISE**

# **1.MEASURE**

# WHAT TO MEASURE?

- Overall page size
- Number of server requests
- The size of the bundled JS resources
- etc.

**WHAT TO MEASURE?**

**PAGE  
LOADING  
TIME**

# WHAT TO MEASURE?

PAGE  
LOADING  
TIME

# WHAT TO MEASURE?

## PAGE LOADING TIME

- DOMContentLoaded Event
- Onload Event
- First Paint
- First Contentful Paint
- First Meaningful Paint
- Largest Contentful Paint
- SpeedIndex

gitlab.com/snippets/1941460

☆

🔒

Incognito

🐙

GitLab

Projects

Groups

Snippets

Help

Search or jump to...

?

Sign in / Register

Snippets > \$1941460

🌐

Authored 2 months ago by

👤

Denys Mishunov

52

JEST component performance

Performance of a component comparing to another implementation

This *should* also support **formatting**

Edited 2 weeks ago

Embed

<script src='http://gitlab.com/snippets/1941460.js'></script>

🔗

📄

test.js

1.22 KIB

🔗

Clone

🔗

📄

📄

📄

1

describe('performance', () => {

2

const simpleViewerB = {

3

extends: SimpleViewer,

4

methods: {

5

shouldInit() {

6

return true;

7

}

8

};

9

};

10

11

const longContent = (numbers = 100) => {

12

let res = '';

13

for (let n = 0; n < 1; ++n) {

14

res += '<span id="LC1">First</span>\n';

15

}

16

return res;

17

};

18

19

const createComponentB = (content = longContent()) => {

20

wrapper = shallowMount(simpleViewerB, {

21

propsData: {

22

content,

23

},

24

});

25

};

■

First Contentful Paint

2.9 s

●

Speed Index

3.2 s

▲

Time to Interactive

10.1 s

▲

First Meaningful Paint

7.7 s

▲

First CPU Idle

10.0 s

▲

Max Potential First Input Delay

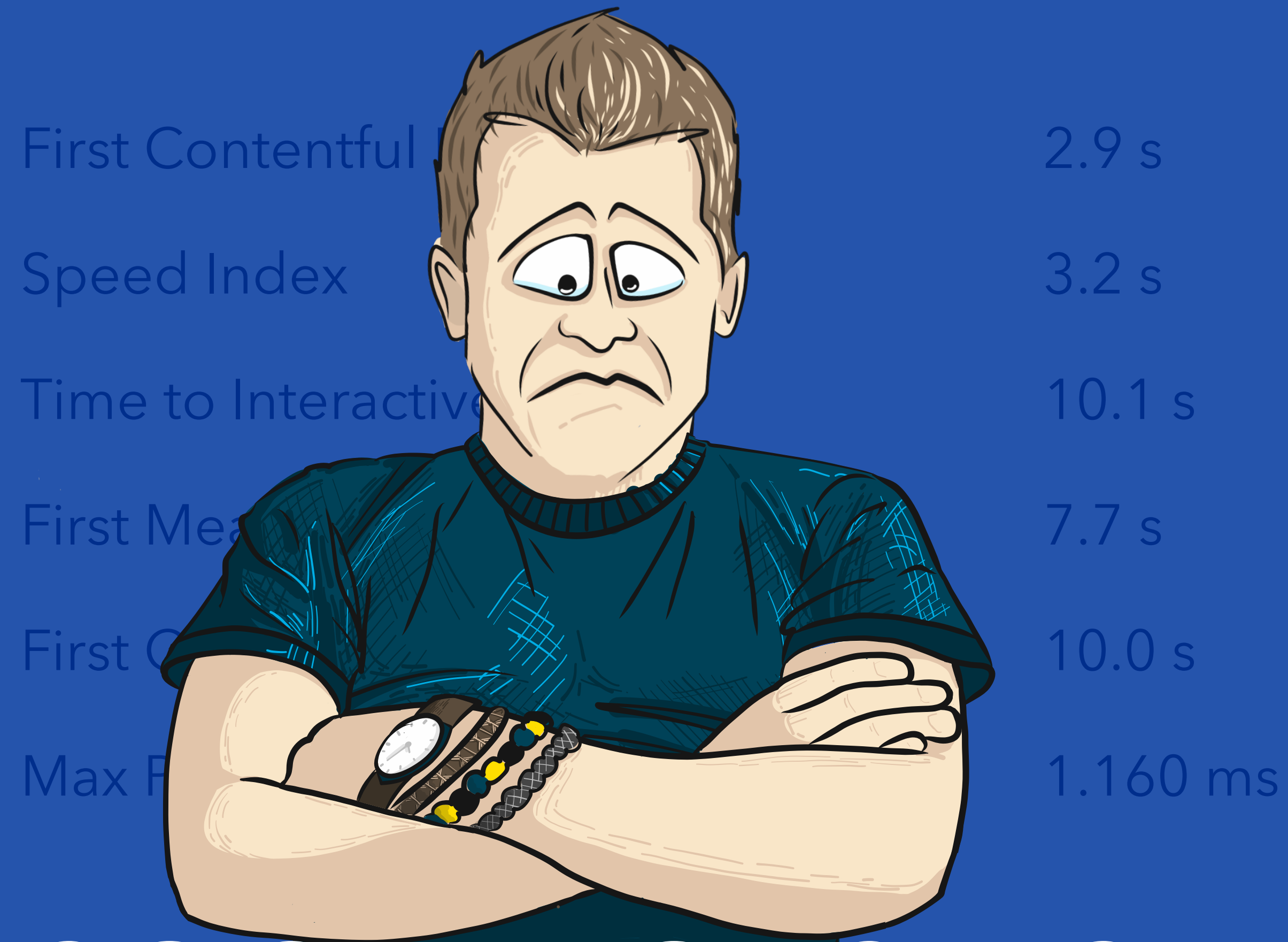
1.160 ms

First Contentful Paint	2.9 s
Speed Index	3.2 s
Time to Interactive	10.1 s
First Meaningful Paint	7.7 s
First CPU Idle	10.0 s
Max Potential First Input Delay	1.160 ms

**Generic**

**Could be fragile for monitoring**

**There are too many**



**HOW SOON DOES USER SEE  
THE SNIPPET?**

# WHAT TO MEASURE?

## PAGE LOADING TIME

- DOMContentLoaded Event
- Onload Event
- First Paint
- First Contentful Paint
- First Meaningful Paint
- Largest Contentful Paint
- SpeedIndex

# WHAT TO MEASURE?

PAGE  
LOADING  
TIME

USER TIMING API

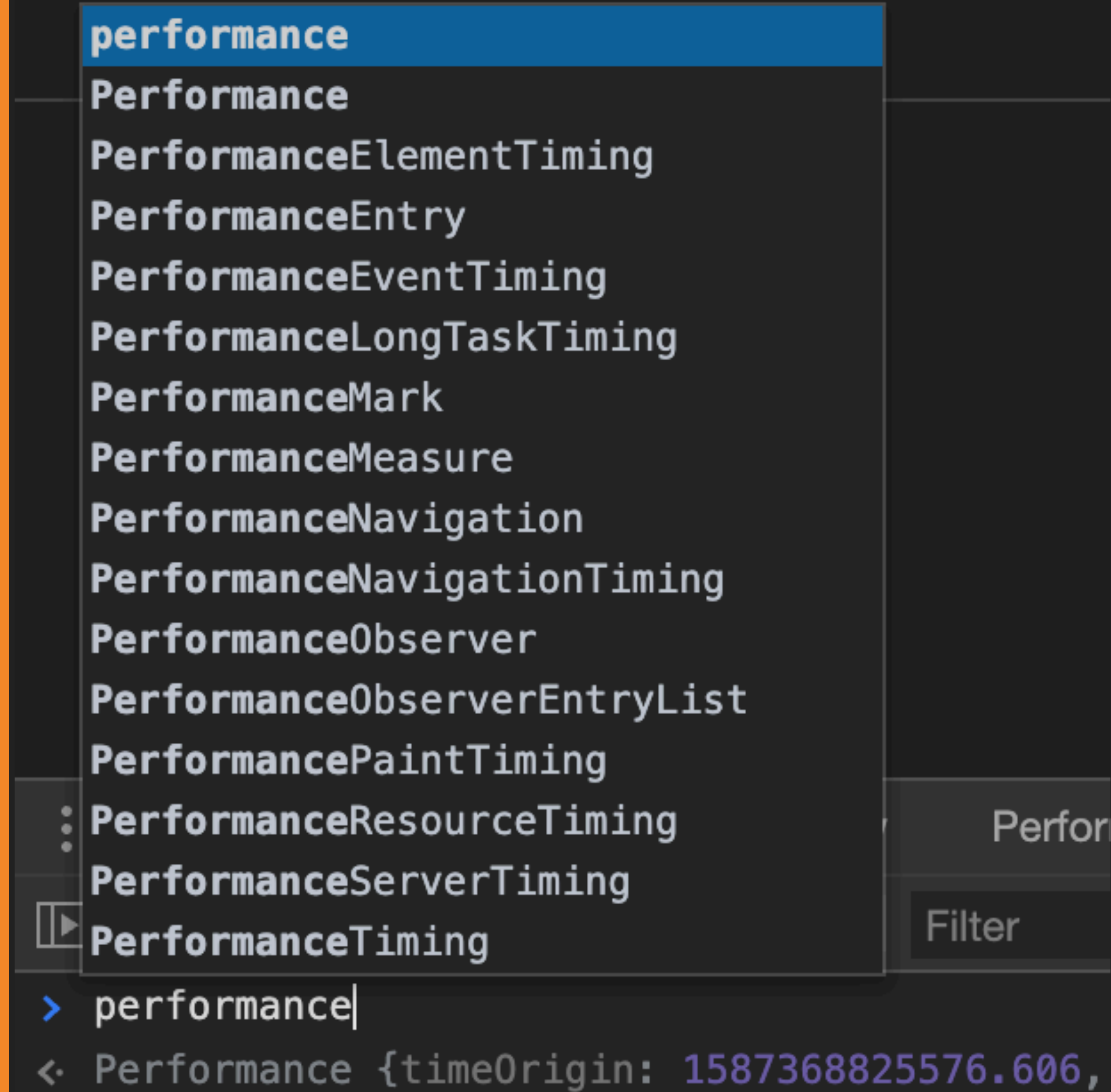
# PERFORMANCE API

<https://developer.mozilla.org/en-US/docs/Web/API/Performance>

# PERFORMANCE API

<https://developer.mozilla.org/en-US/docs/Web/API/Performance>

## Performance Timeline API



# PERFORMANCE API

<https://developer.mozilla.org/en-US/docs/Web/API/Performance>

## Performance Timeline API

## Navigation Timing API\*

The image shows two screenshots of the Chrome DevTools interface. The top screenshot displays the Performance tab with a timeline view showing a single navigation event. The bottom screenshot displays the Console tab with the expanded `performance.timing` and `performance.navigation` objects.

**Top Screenshot (Performance Tab):**

- Navigation Start: 1587368825578
- Connect End: 1587368825579
- Connect Start: 1587368825579
- DOM Complete: 1587368833413
- DOM Content Loaded Event End: 1587368833408
- DOM Content Loaded Event Start: 1587368833380
- DOM Interactive: 1587368827901
- DOM Loading: 1587368826171
- Domain Lookup End: 1587368825579
- Domain Lookup Start: 1587368825579
- Fetch Start: 1587368825579
- Load Event End: 1587368833414
- Load Event Start: 1587368833413
- Navigation Start: 1587368825578
- Redirect End: 0
- Redirect Start: 0
- Request Start: 1587368825580
- Response End: 1587368826199
- Response Start: 1587368826086
- Secure Connection Start: 0
- Unload Event End: 1587368826166
- Unload Event Start: 1587368826166

**Bottom Screenshot (Console Tab):**

- `performance.timing` object expanded, showing the same timing data as the top screenshot.
- `performance.navigation` object expanded, showing:
  - `type`: 1
  - `redirectCount`: 0

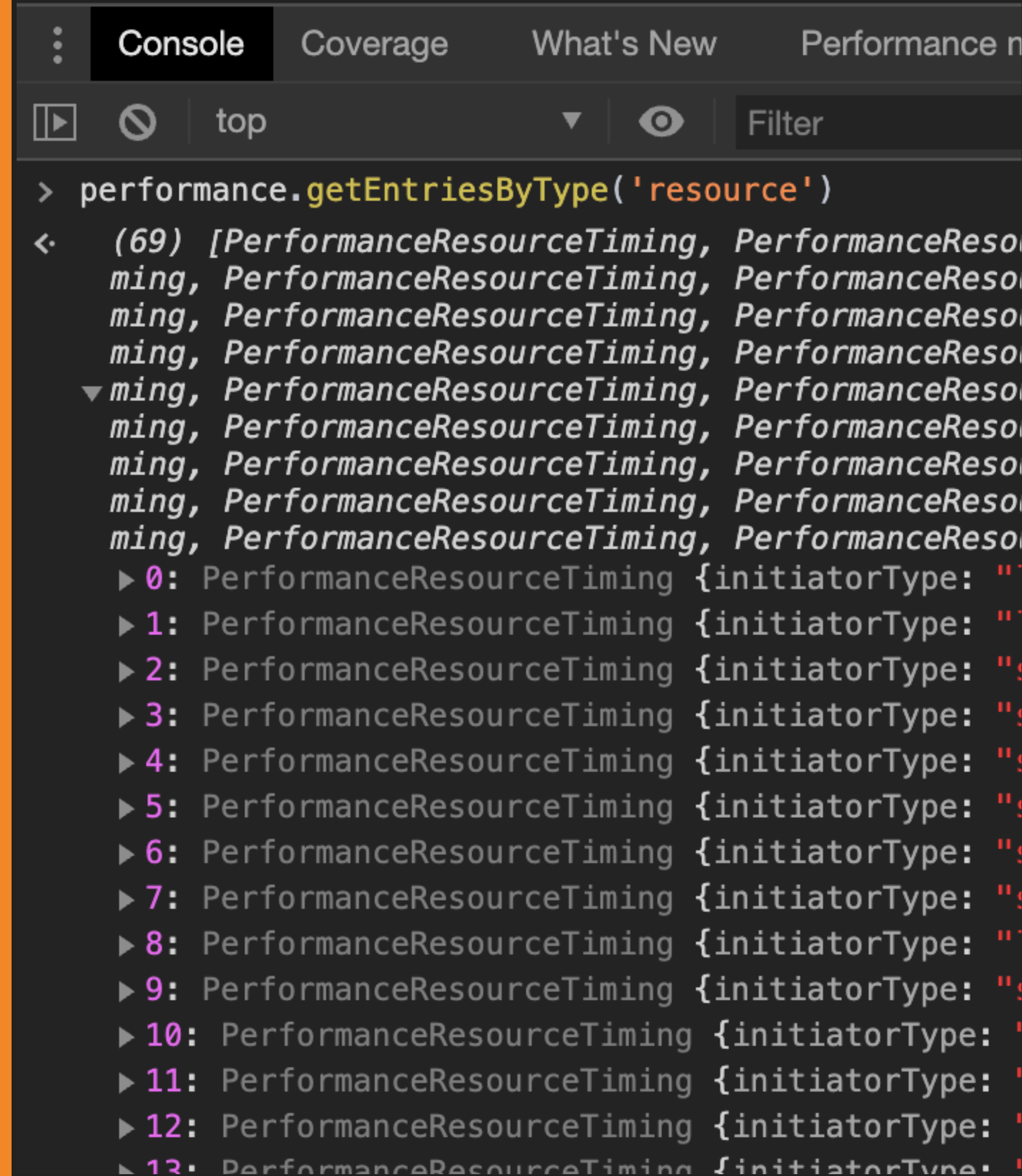
# PERFORMANCE API

# Performance Timeline API

# Navigation Timing API\*

# Resource Timing API

<https://mzl.la/2Kg4oIT>



# PERFORMANCE API

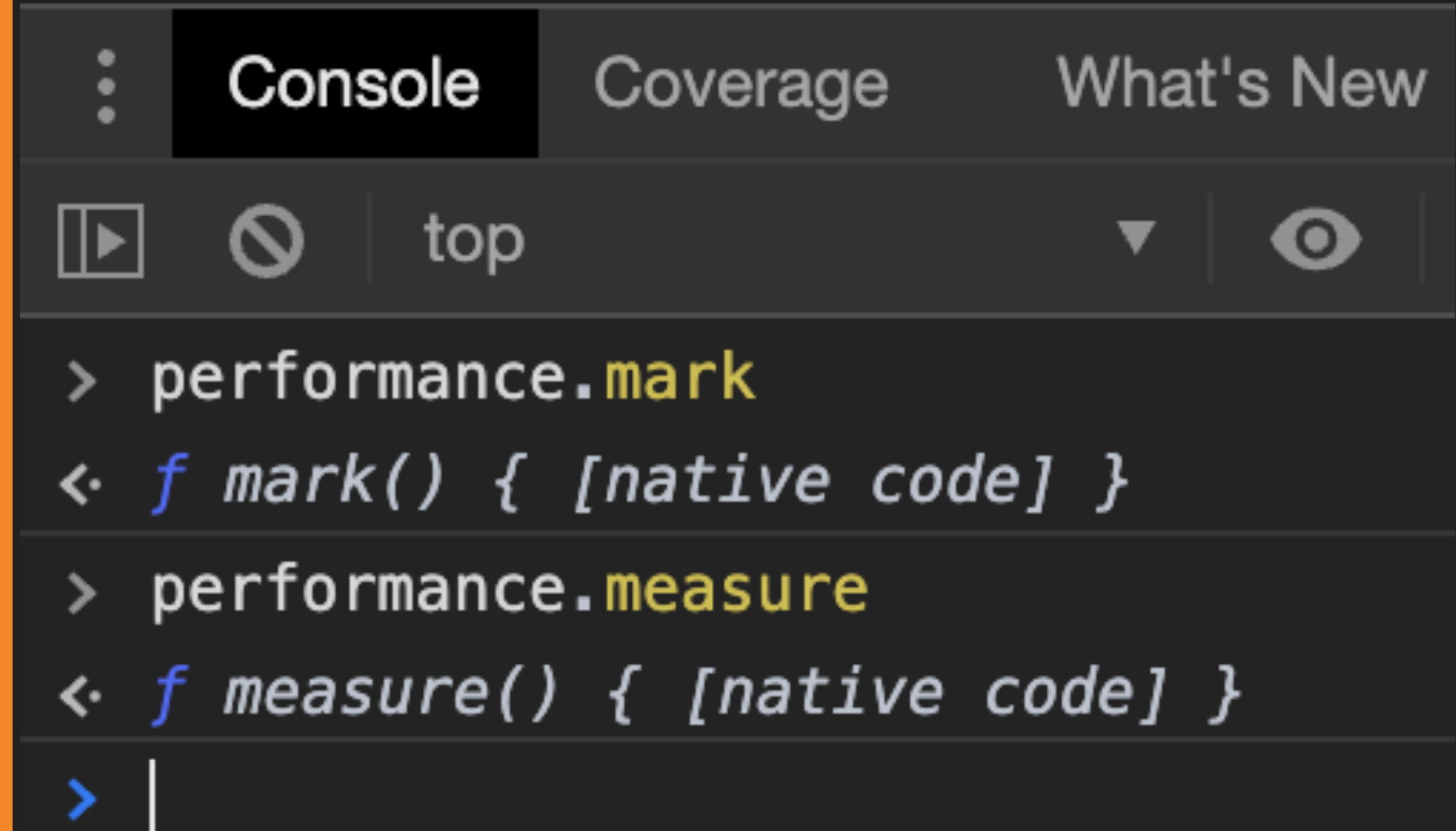
<https://developer.mozilla.org/en-US/docs/Web/API/Performance>

Performance Timeline API

Navigation Timing API\*

Resource Timing API

User Timing API



```
⋮ Console Coverage What's New
▶ 🔍 top ▼ 👁
> performance.mark
< f mark() { [native code] }
> performance.measure
< f measure() { [native code] }
> |
```

# WHAT TO MEASURE?

PAGE  
LOADING  
TIME

USER TIMING API

# SHOW ME HOW

PRACTICAL EXAMPLE



JEST component performance

localhost:3000/snippets/72

dmishunov.local21ms / 16 pg6ms / 1 gitaly105ms / 52 redis917 | 1600 | 4676 / 55 totaltrace+Downloadsnippets/72

GitLabProjectsGroupsMore+Search or jump to...8314

Snippets > \$72

Author2 days ago by AdministratorEditDeleteNew snippet

JEST component performance

Performance of a component comparing to another implementation

Edited 16 hours ago

Embed<script src='http://127.0.0.1:3000/snippets/72.js'></script>

spec.js1.22 KiBClone

```
1 describe('performance', () => {
2   const simpleViewerB = {
3     extends: SimpleViewer,
4     methods: {
5       scrollToLine(hash) {
6         return true;
7       }
8     }
9   };
10
11   const longContent = (numbers = 100) => {
12     let res = '';
13     for (let n = 0, l = numbers; n < l; ++n) {
14       res += '<span id="LC1">First</span>\n';
15     }
16     return res;
17   };
18
19   const createComponentB = (content = longContent()) => {
20     wrapper = shallowMount(simpleViewerB, {
21       propsData: {
22         content,
```

Performance

localhost #1ScreenshotsMemory

1000 ms2000 ms3000 ms4000 ms5000 ms6000 ms7000 ms8000 msFPSCPUNET

Network72 (localhost)apli...dark...main.chunk....

Frames608.7 ms832.7 ms2274.6 ms

Interactions

TimingsFPFCPDCLLCPFMP

Main — http://localhost:3000/snippets/72

TaskEvaluate Script(anonymous)webpackJs...CallbackcheckDef...dModules

SummaryBottom-UpCall TreeEvent Log

Range: 0 – 6.84 s

67 ms Loading3062 ms Scripting153 ms Rendering36 ms Painting1697 ms System1827 ms Idle

6841 ms

Snippets x

Author

Edit Delete New snippet

## JEST component performance

### Performance of a component comparing to another implementation

Edited 16 hours ago

Embed  `<script src='http://127.0.0.1:3000/snippets/72.js'></script>` 

JS

spec.js 1.22 KiB



Clone   

```

1 describe('performance', () => {
2   const simpleViewerB = {
3     extends: SimpleViewer,
4     methods: {
5       scrollToLine(hash) {
6         return true;
7       }
8     }
9   };
10
11   const longContent = (numbers = 100) => {
12     let res = '';
13     for (let n = 0, l = numbers; n < l; ++n) {
14       res += '<span id="LC1">First</span>\n';
15     }
16     return res;
17   };
18
19   const createComponentB = (content = longContent()) => {
20     wrapper = shallowMount(simpleViewerB, {
21       propsData: {
22         content,

```

🔍 Filter components

```

▼ <Root>
  ▼ <Show>
    ▶ <SnippetHeader>
    ▶ <SnippetTitle>
    ▼ <SnippetBlob>
      ▶ <BlobEmbeddable>
      ▶ <BlobHeader>
      ▼ <BlobContent>
        ▼ <SimpleViewer> = $vm0

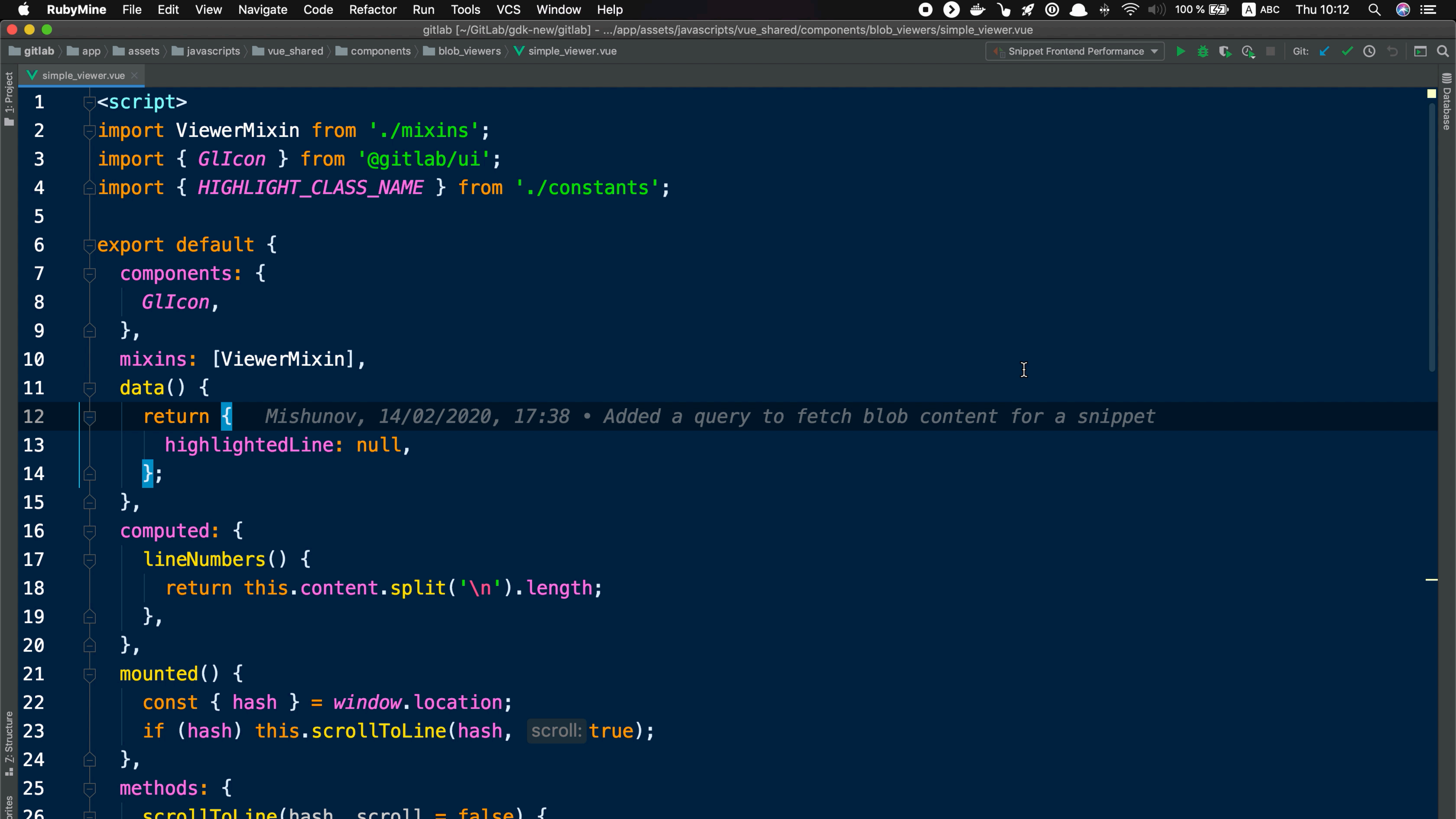
```

```

▼ data
  ▼ $apolloData: Object
    ► data: Object (empty)
      loading: 0
    ► queries: Object (empty)
    highlightedLine: null

```

```
▼ computed
  lineNumbers: 53
```





RubyMineFileEditViewNavigateCodeRefactorRunToolsVCSWindowHelp

gitlab [~/GitLab/gdk-new/gitlab] - .../app/assets/javascripts/vue\_shared/components/blob\_viewers/simple\_viewer.vue

gitlab > app > assets > javascripts > vue\_shared > components > blob\_viewers > simple\_viewer.vue

Snippet Frontend Performance

Git: ✓✓🕒🔄🔍

1: Project

Database

Z: Structure

ories

simple\_viewer.vue

1<script>

2import ViewerMixin from './mixins';

3import { GIcon } from '@gitlab/ui';

4import { HIGHLIGHT\_CLASS\_NAME } from './constants';

5

6export default {

7  components: {

8    GIcon,

9  },

10  mixins: [ViewerMixin],

11  data() {

12    return {

13      highlightedLine: null,

14    };

15  },

16  computed: {

17    lineNumbers() {

18      return this.content.split('\n').length;

19    },

20  },

21  mounted() {

22    const { hash } = window.location;

23    if (hash) this.scrollToLine(hash, scroll: true);

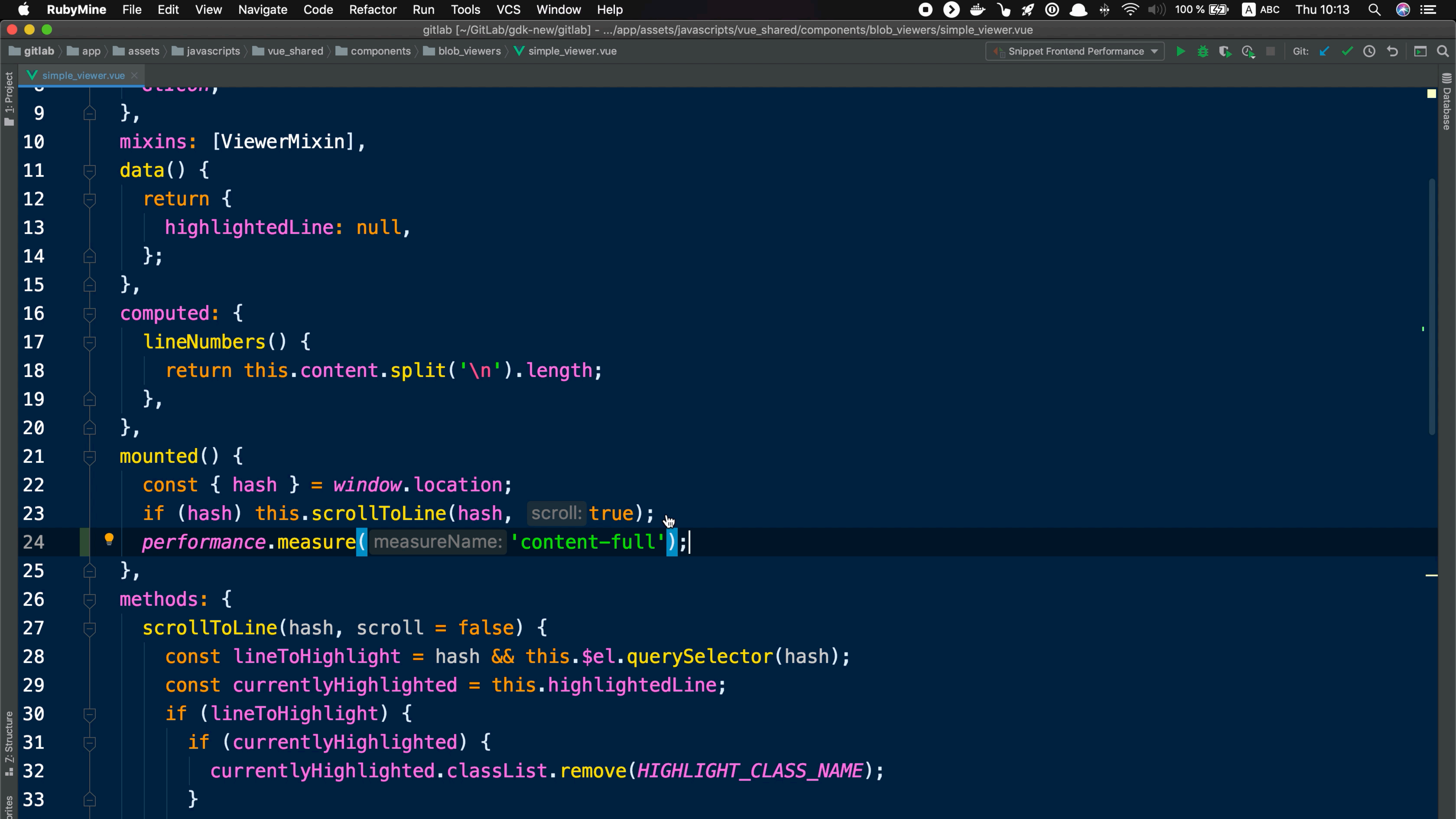
24  },

25  methods: {

26    scrollToLine(hash, scroll = false) {

Mishunov, 14/02/2020, 17:38 • Added a query to fetch blob content for a snippet

performance.measure(MEASURE\_NAME, START\_MARK, END\_MARK);



JEST component performance

localhost:3000/snippets/72

dmishunov.local174ms / 37 pg49ms / 1 gitaly756ms / 132 redis9154 | 10692 | 15872 / 57 totaltrace+Downloadsnippets/72

GitLabProjectsGroupsMore+Search or jump to...8314?

Snippets > \$72

Author2 days ago by AdministratorEditDeleteNew snippet

# JEST component performance

Performance of a component comparing to another implementation

Edited 17 hours ago

Embed<script src='http://127.0.0.1:3000/snippets/72.js'></script>

spec.js1.22 KiBClone

```
1 describe('performance', () => {
2   const simpleViewerB = {
3     extends: SimpleViewer,
4     methods: {
5       scrollToLine(hash) {
6         return true;
7       }
8     }
9   };
10
11   const longContent = (numbers = 100) => {
12     let res = '';
13     for (let n = 0, l = numbers; n < l; ++n) {
14       res += '<span id="LC1">First</span>\n';
15     }
16     return res;
17   };
18
19   const createComponentB = (content = longContent()) => {
20     wrapper = shallowMount(simpleViewerB, {
21       propsData: {
22         content,
```

Performance

localhost #1ScreenshotsMemory

2000 ms4000 ms6000 ms8000 ms10000 ms12000 ms14000 ms16000 ms18000 ms20000 ms

Network

72 (localhost)

apli...per...d...main.chunk.js...

Frames

2657.7 ms

Interactions

Timings

content-full

Main — http://localhost:3000/snippets/72

Task

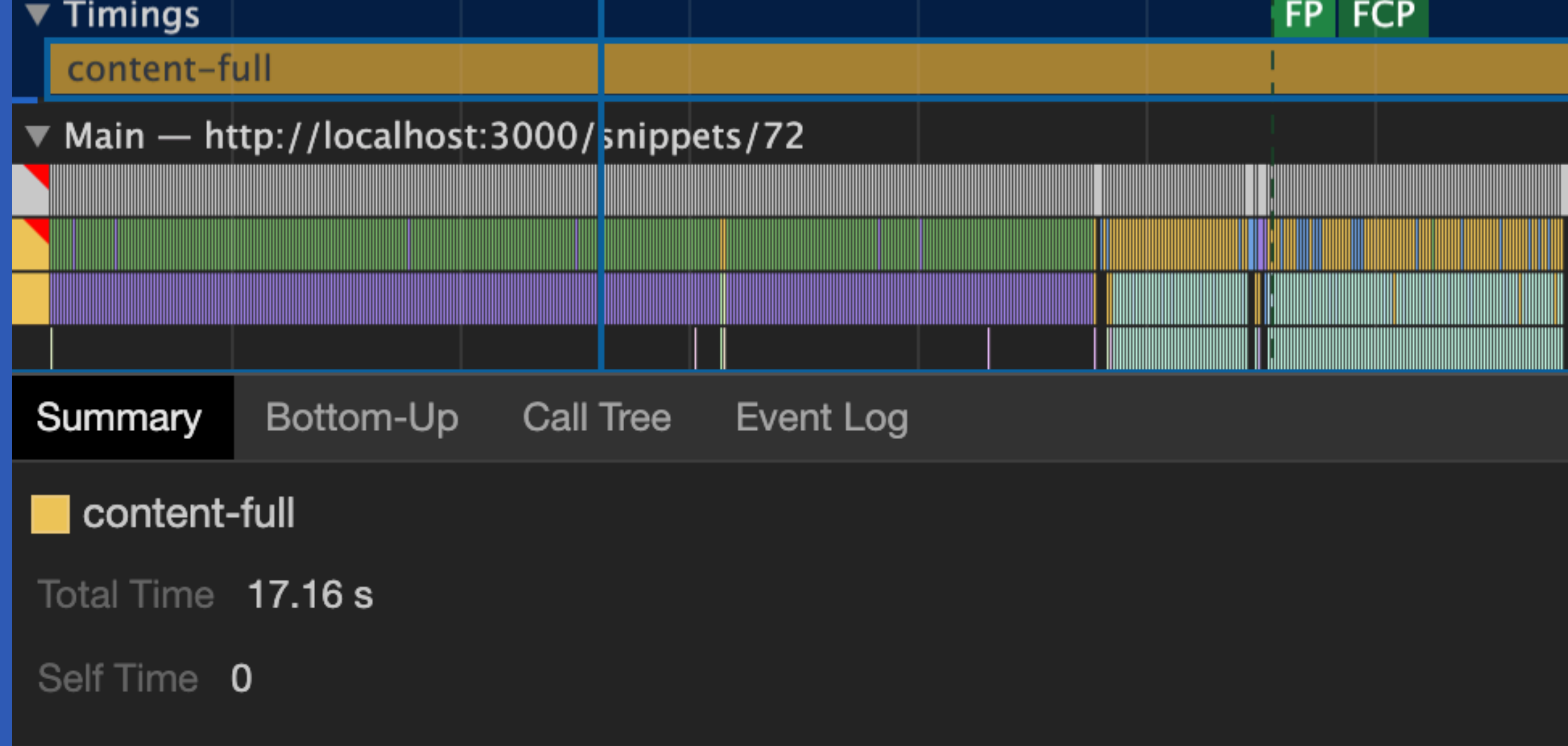
Eval...ipt(an...us)web...ck

Summary

content-full


Total Time 17.16 s

Self Time 0



**IS VUE THE  
PROBLEM?**

Snippets > **\$72**

🌐 Authored 2 days ago by  Administrator

Edit


Delete

New snippet

## JEST component performance

### Performance of a component comparing to another implementation

Edited 17 hours ago

Embed 

```
<script src='http://127.0.0.1:3000/snippets/72.js'></script>
```



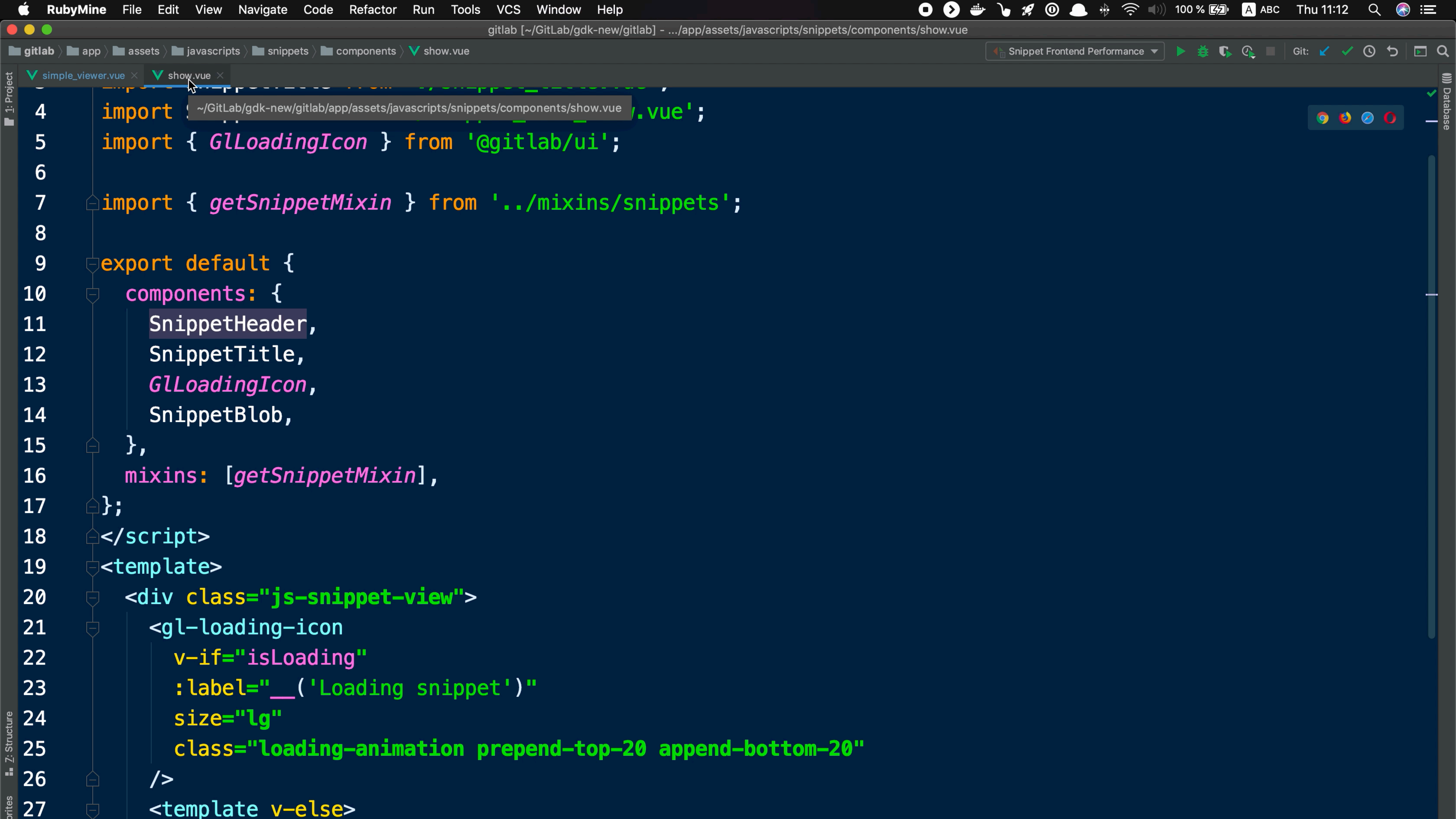
JS spec.js 1.22 KiB

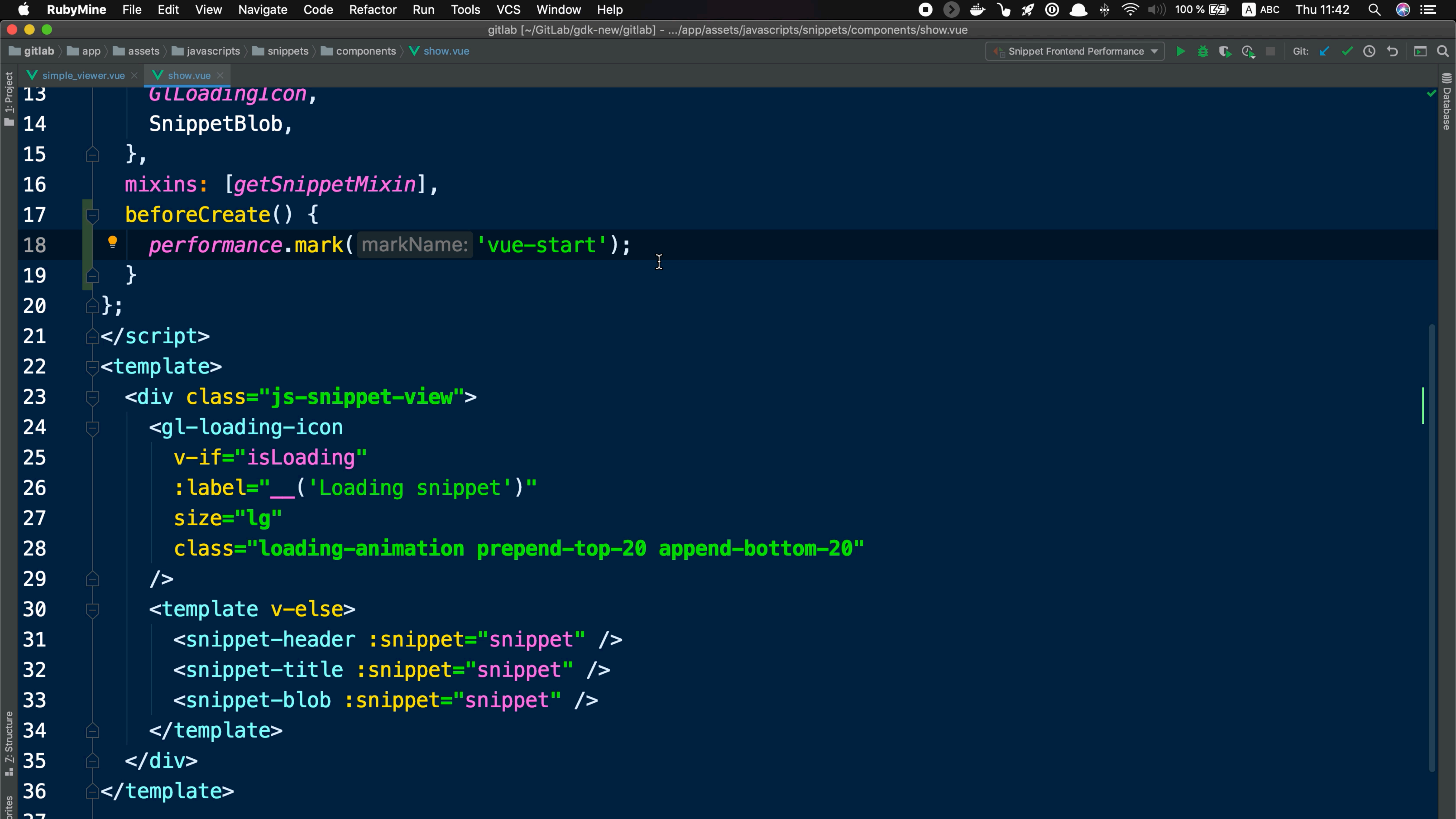
Clone 

```

1 describe('performance', () => {
2   const simpleViewerB = {
3     extends: SimpleViewer,
4     methods: {
5       scrollToLine(hash) {
6         return true;
7       }
8     }
9   };
10
11   const longContent = (numbers = 100) => {
12     let res = '';
13     for (let n = 0, l = numbers; n < l; ++n) {
14       res += '<span id="LC1">First</span>\n';
15     }
16     return res;
17   };
18
19   const createComponentB = (content = longContent()) => {
20     wrapper = shallowMount(simpleViewerB, {
21       propsData: {
22         content,

```





JEST component performance

localhost:3000/snippets/72

dmishunov.local36ms / 17 pg10ms / 1 gitaly145ms / 54 redis1271 | 1991 | 5398 / 26 totaltrace+Downloadsnippets/72

GitLabProjectsGroupsMoreSearch or jump to...

Snippets > \$72

Authored 3 days ago by AdministratorEditDeleteNew snippet

# JEST component performance

Performance of a component comparing to another implementation

Edited 18 hours ago

Embed <script src='http://127.0.0.1:3000/snippets/72.js'></script>

spec.js1.22 KiBClone

```
1 describe('performance', () => {
2   const simpleViewerB = {
3     extends: SimpleViewer,
4     methods: {
5       scrollToLine(hash) {
6         return true;
7       }
8     }
9   };
10
11   const longContent = (numbers = 100) => {
12     let res = '';
13     for (let n = 0, l = numbers; n < l; ++n) {
14       res += '<span id="LC1">First</span>\n';
15     }
16     return res;
17   };
18
19   const createComponentB = most probably due to the development env
20   wrapper = shallowMount(simpleViewerB, {
21     propsData: {
22       content,
```

75% waiting  
to bootstrap the Vue application\*

ElementsConsoleSourcesNetworkPerformanceMemory

(no recordings)ScreenshotsMemory

Status Processing profile...  
Received  
Stop

# USER TIMING API




# USER TIMING API

performance.mark(START\_MARK);




# USER TIMING API

`performance.mark(START_MARK);`



A horizontal orange line represents a timeline. A vertical yellow line segment intersects the orange line at the left end, marking the start of a performance measurement.

`performance.mark(END_MARK);`



A horizontal orange line represents a timeline. A vertical yellow line segment intersects the orange line at the right end, marking the end of a performance measurement.

# USER TIMING API

`performance.mark(START_MARK);`

`performance.mark(END_MARK);`

`performance.measure(MEASURE_NAME, START_MARK, END_MARK);`

# USER TIMING API

`performance.mark(START_MARK);`

---

`performance.measure(MEASURE_NAME, START_MARK);`

# USER TIMING API

---

```
performance.measure(MEASURE_NAME);
```

# USER TIMING API

## **CROSS-COMPONENT**

*Marks can be placed and measured anywhere on the page*

## **NOT VUE-SPECIFIC**

*Can be added to any JS*

---

# DEVTOOLS ARE **NOT** **SCALABLE**

Common sense



# MEASURE • MONITOR • OPTIMISE

Page Loading Time  
User Timing API

## **2. MONITOR**

---

# MONITOR

---

# **MONITOR**

## **ANALYTICS**

## **PERFORMANCE**

# MONITOR

## ANALYTICS TOOLS

Google Analytics

Snowplow 

etc.

### Site Speed Overview

Email Export Add to Dashboard Shortcut

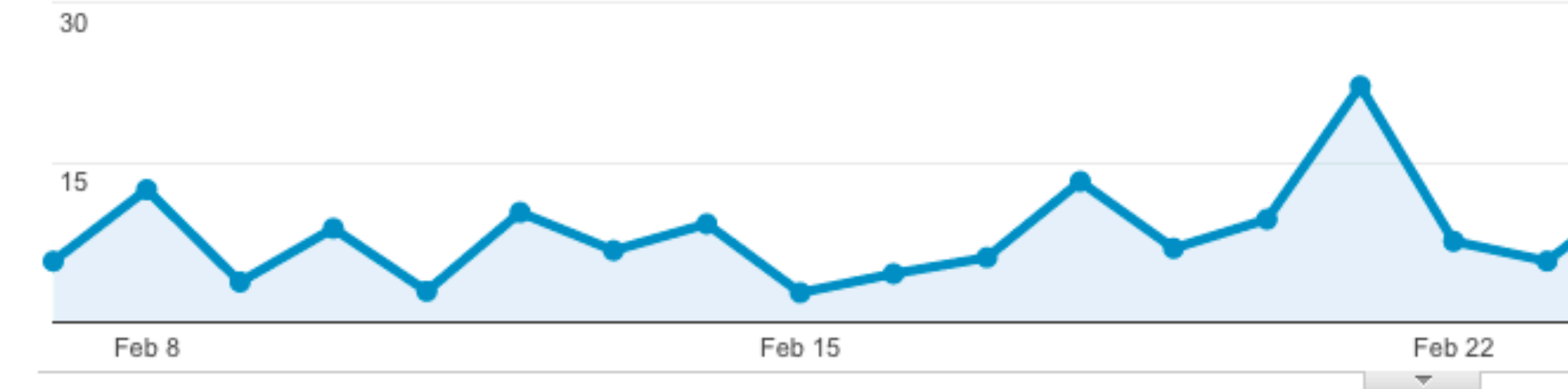
All Users  
100.00% Pageviews

+ Add Segment

#### Overview

Avg. Page Load Time (sec) vs. [Select a metric](#)

Avg. Page Load Time (sec)



1,361 of pageviews sent page load sample

Avg. Page Load Time (sec)

6.13



Avg. Redirection Time (sec)

0.36



Avg. Domain Lookup Time (sec)

0.02



Avg. Server Response Time (sec)

0.03



#### Site Speed

Browser

Country

Page

#### Browser

1. [Firefox](#)
2. [Internet Explorer](#)
3. [Opera](#)
4. [Safari \(in-app\)](#)
5. [Chrome](#)
6. [Safari](#)
7. [Android Browser](#)

# MONITOR

## PERFORMANCE TOOLS

Webpagetest

Pingdom

sitespeed.io 

SpeedCurve

etc.

HOMETEST RESULTTEST HISTORYFORUMSDOCUMENTATIONABOUT

Web Page Performance Test for

<https://gitlab.com/snippets/1941460>

From: Amsterdam, NL - MeasureWorks - Chrome - 3GFast

19/04/2020, 16:46:08

Need help improving?

B

First Byte Time

A

Keep-alive Enabled

A

Compress Transfer

A

Compress Images

B

Cache static content

✓

Effective use of CDN

SummaryDetailsPerformance ReviewContent BreakdownDomainsProcessing BreakdownScreenshotImage AnalysisRequest Map

Tester: wpt-ubuntu-s-2vcpu-4gb-ams3-01-206.189.108.106

First View only

Test runs: 3

Re-run the test

View JSON result

Raw page data - Raw object data

Export HTTP Archive (.har)

View Test Log



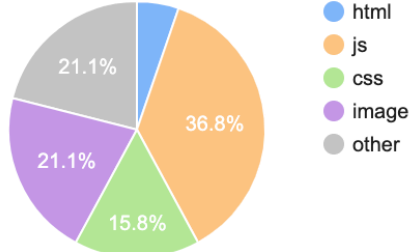
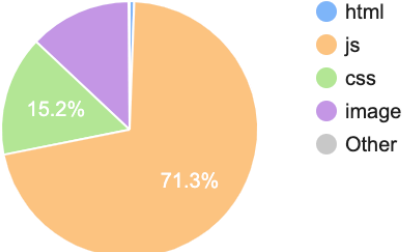
Performance Results (Median Run - SpeedIndex)

	Load Time	First Byte	Start Render	First Contentful Paint	Speed Index	Last Painted Hero	First CPU Idle	Document Complete			Fully Loaded			
								Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View (Run 1)	8.624s	0.970s	3.100s	3.082s	5.084s	9.400s	8.559s	8.624s	13	1,147 KB	11.212s	19	1,287 KB	\$\$\$-:


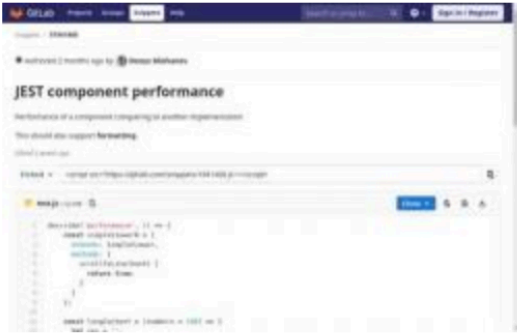
Plot Full Results

Test Results



Run 1:

	Waterfall	Screenshot	Video
<div>First View (8.624s)</div> <div><a href="#">Timeline (view)</a></div> <div><a href="#">Processing Breakdown</a></div> <div><a href="#">Trace (view)</a></div>			<div><a href="#">Filmstrip View</a></div> <div><a href="#">Watch Video</a></div>
<div><a href="#">Content Breakdown</a></div>	<div><div>Requests</div><div><div>html</div><div>js</div><div>css</div><div>image</div><div>other</div></div><div><div>Bytes</div><div><div>html</div><div>js</div><div>css</div><div>image</div><div>Other</div></div></div></div>		

Run 2:

	Waterfall	Screenshot	Video
<div>First View (8.672s)</div> <div><a href="#">Timeline (view)</a></div> <div><a href="#">Processing Breakdown</a></div> <div><a href="#">Trace (view)</a></div>			<div><a href="#">Filmstrip View</a></div> <div><a href="#">Watch Video</a></div>

Run 3:

	Waterfall	Screenshot	Video
<div>First View (8.681s)</div> <div><a href="#">Timeline (view)</a></div> <div><a href="#">Processing Breakdown</a></div> <div><a href="#">Trace (view)</a></div>			<div><a href="#">Filmstrip View</a></div> <div><a href="#">Watch Video</a></div>

---

# MONITOR

ANALYTICS

PERFORMANCE

**SNOWPLOW**

**SITESPEED.IO**

# MONITOR SNOWPLOW

<https://docs.gitlab.com/ee/telemetry/index.html>

- Clicking links or buttons.
- Submitting forms.
- **Other typically interface-driven actions**

# MONITOR SNOWPLOW

<https://docs.gitlab.com/ee/telemetry/index.html>

- Event tracking: <https://docs.gitlab.com/ee/telemetry/index.html>
- Snowplow tracking guide: <https://docs.gitlab.com/ee/telemetry/snowplow.html#frontend-tracking>
- Enable Snowplow tracking in **Admin Area > Settings > Integrations**
- [\*\*Add tracking to your code\*\*](#)



# MONITOR

PERFORMANCE TOOLS

RubyMineFileEditViewNavigateCodeRefactorRunToolsVCSWindowHelp

gitlab [~/GitLab/gdk-new/gitlab] - .../app/assets/javascripts/vue\_shared/components/blob\_viewers/simple\_viewer.vue

gitlab > app > assets > javascripts > vue\_shared > components > blob\_viewers > simple\_viewer.vue

Snippet Frontend Performance

Git:

1: Project

Database

simple\_viewer.vue x show.vue x

1<script>

2import ViewerMixin from './mixins';

3import { GLIcon } from '@gitlab/ui';

4import { HIGHLIGHT\_CLASS\_NAME } from './constants';

5

6export default {

7  components: {

8    GLIcon,

9  },

10  mixins: [ViewerMixin],

11  data() {

12    return {

13      highlightedLine: null,

14    };

15  },

16  computed: {

17    lineNumbers() {

18      return this.content.split('\n').length;

19    },

20  },

21  mounted() {

22    const { hash } = window.location;

23    if (hash) this.scrollToLine(hash, true);

24    performance.measure( measureName: 'content-full');

25    performance.measure( measureName: 'content-within-vue', startMark: 'vue-start');

26  }

JEST component performance

localhost:3000/snippets/72

dmishunov.local34ms / 16 pg3ms / 1 gitaly166ms / 33 redis1078 | 1872 | 5939 / 28 totaltrace+Downloadsnippets/72

GitLab

ProjectsGroupsMore

Search or jump to...

8314

Snippets > \$72

Author 3 days ago by Administrator

EditDeleteNew snippet

# JEST component performance

Performance of a component comparing to another implementation

Edited 23 hours ago

Embed<script src='http://127.0.0.1:3000/snippets/72.js'></script>

spec.js1.22 KiB

Clone

```
1 describe('performance', () => {
2   const simpleViewerB = {
3     extends: SimpleViewer,
4     methods: {
5       scrollToLine(hash) {
6         return true;
7       }
8     }
9   };
10
11   const longContent = (numbers = 100) => {
12     let res = '';
13     for (let n = 0, l = numbers; n < l; ++n) {
14       res += '<span id="LC1">First</span>\n';
15     }
16     return res;
17   };
18
19   const createComponentB = (content = longContent()) => {
20     wrapper = shallowMount(simpleViewerB, {
21       propsData: {
22         content,
23       },
```

poplin data

Clear Events

Clear Schema Cache

Import Bad Rows

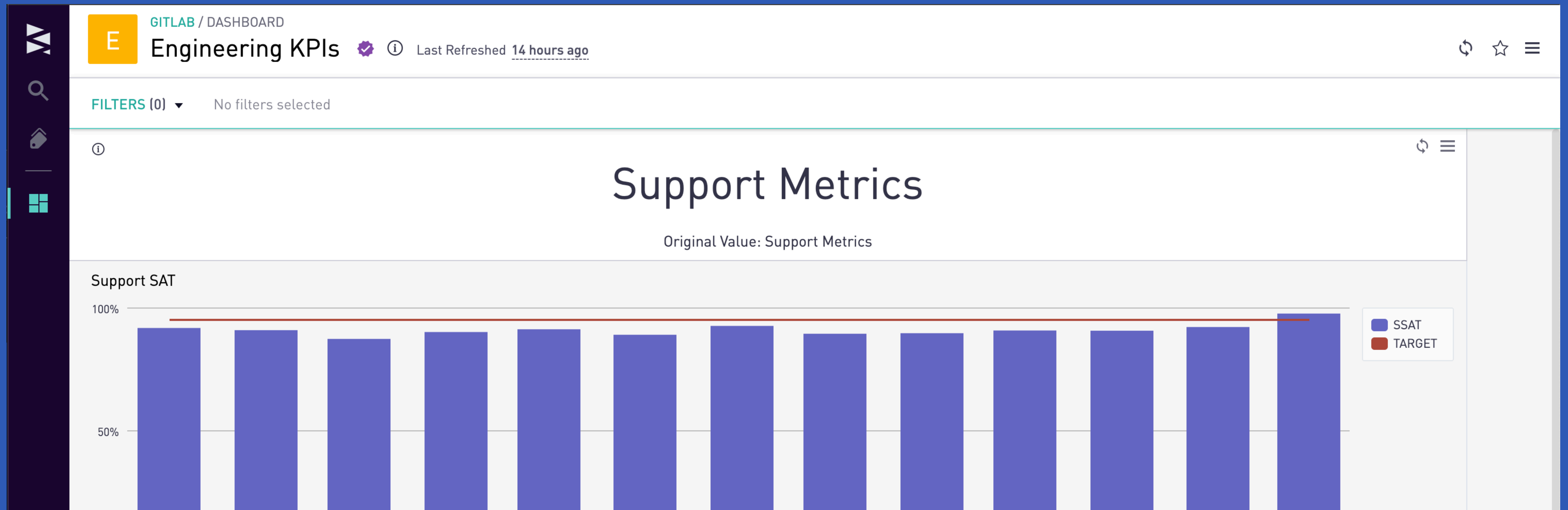
Stream Live Data

Im

Filter

# MONITOR SNOWPLOW → SISENSE

(ex-Periscope)



# MONITOR SNOWPLOW

## PROS:

- Monitors *exactly* what you send to it

## CONS:

- Monitors *exactly* what you send to it
- Metrics won't work with DNT setting
- Requires analytics-specific code

---

# MONITOR

ANALYTICS

PERFORMANCE

SNOWPLOW

SITESPEED.IO

# MONITOR SITESPEED

<https://gitlab.com/gitlab-org/frontend/sitespeed-measurement-setup>

GitLab

Next

Projects

Groups

More

+

Search or jump to...

29

6

14

S

GitLab.org > Frontend > sitespeed-measurement-setup > Details

S

sitespeed-measurement-setup

Project ID: 14213955

Unstar

4

Fork

0

65 Commits

2 Branches

0 Tags

717 KB Files

717 KB Storage

Setup to measure performance on Gitlab websites (.com, dev.) through sitespeed.io and report to Grafana

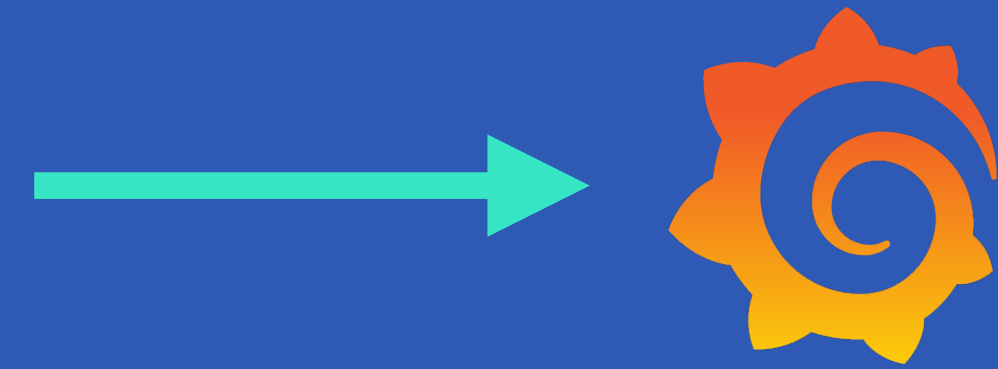
Auto DevOps

It will automatically build, test, and deploy your application based on a predefined CI/CD configuration.

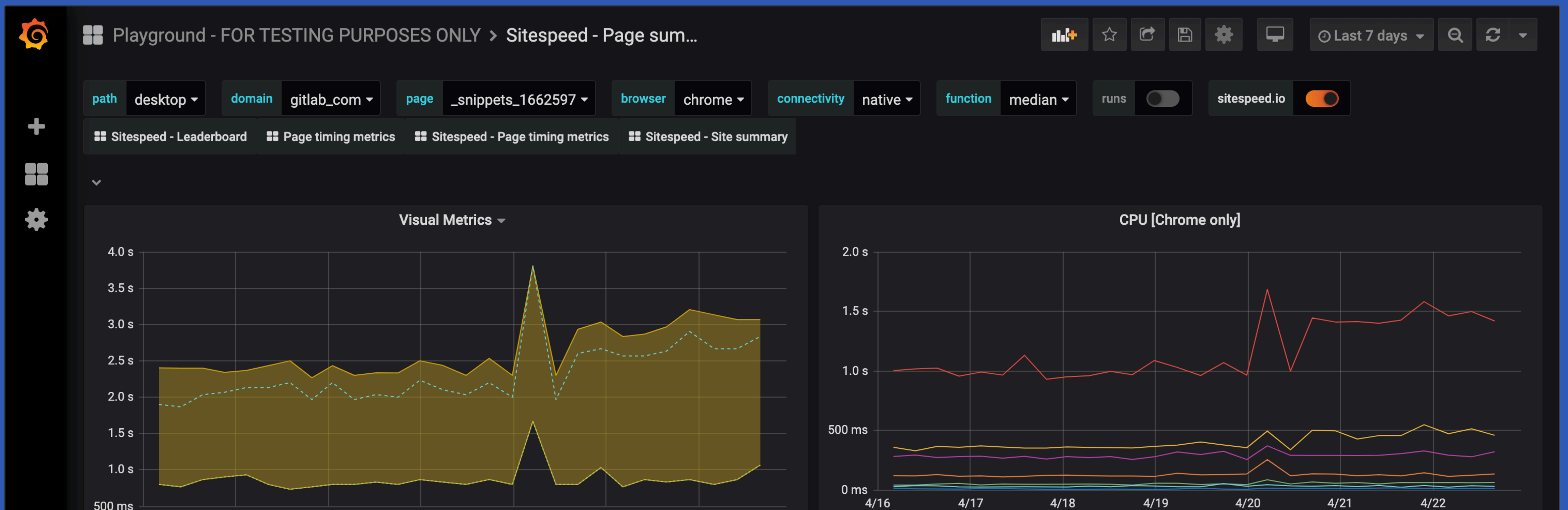
Learn more in the [Auto DevOps documentation](#)

Enable in settings

# MONITOR SITESPEED



Grafana



# MONITOR **SITESPEED**

## PROS:

- Dedicated performance tool
- No additional code required

## CONS:

- Setting up Docker to run against localhost or local host (defined in /etc/hosts) is a **nightmare**

---

**MONITOR**

ANALYTICS

PERFORMANCE

**RETROSPECTIVE**

---

**MONITOR**

**PROACTIVE**

**Practical part**

```
context 'Frontend Performance' do
  let(:file_name) { 'popen.rb' }
  let(:content) { project.repository.blob_at('master', 'files/ruby/popen.rb').data }

  before do
    stub_feature_flags(snippets_vue: true)
    visit snippet_path(snippet)

    # wait_for_requests
    sleep 5
  end

  it 'starts rendering snippet within 0.5 seconds +-20% percent' do
    expect(page.evaluate_script('window.performance.getEntriesByName("vue-start")[0].startTime/1000')).to be_within(0.125).of(0.5)
  end

  it 'renders full snippet within 2 seconds +-20% percent' do
    expect(page.evaluate_script('window.performance.getEntriesByName("content-full")[0].duration/1000')).to be_within(0.5).of(2.0)
  end
end
```

# MEASURE • MONITOR • OPTIMISE

Page Loading Time  
User Timing API

Sitespeed  
Snowplow

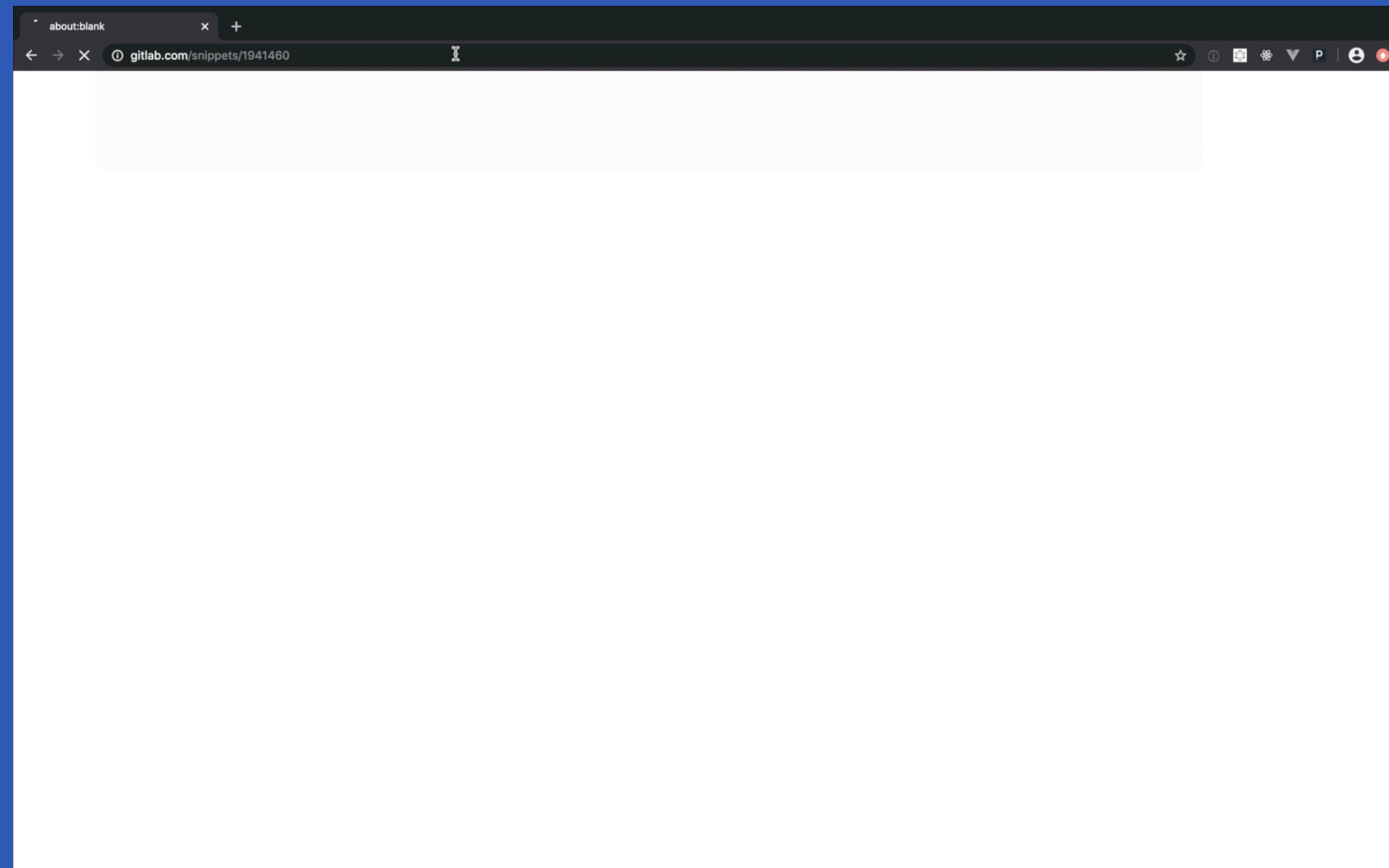
# **3. OPTIMISE**

**IS IT NEEDED?**

**IS IT NEEDED?**

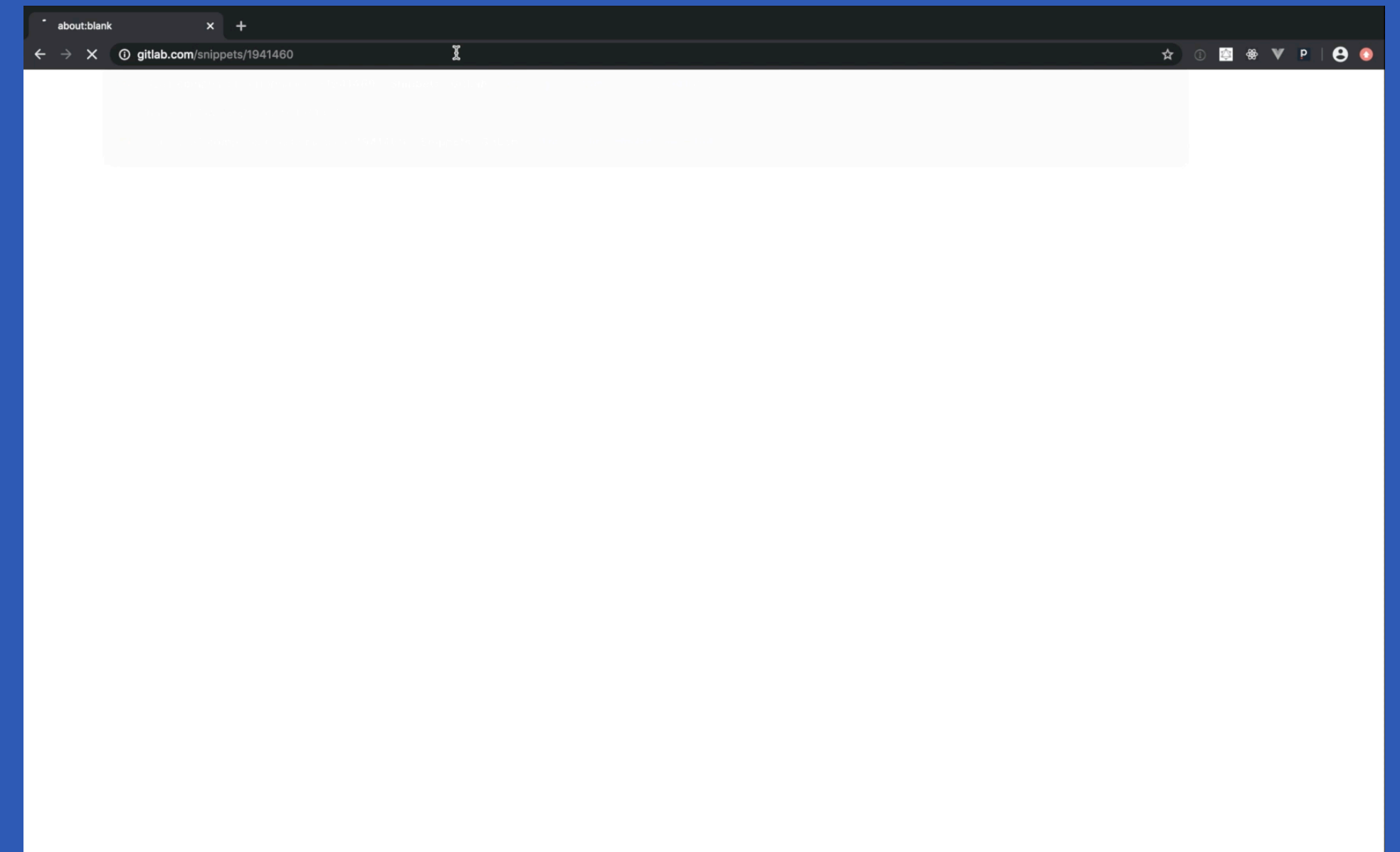
WATCH CLOSELY & GET  
READY

# OPTION #1



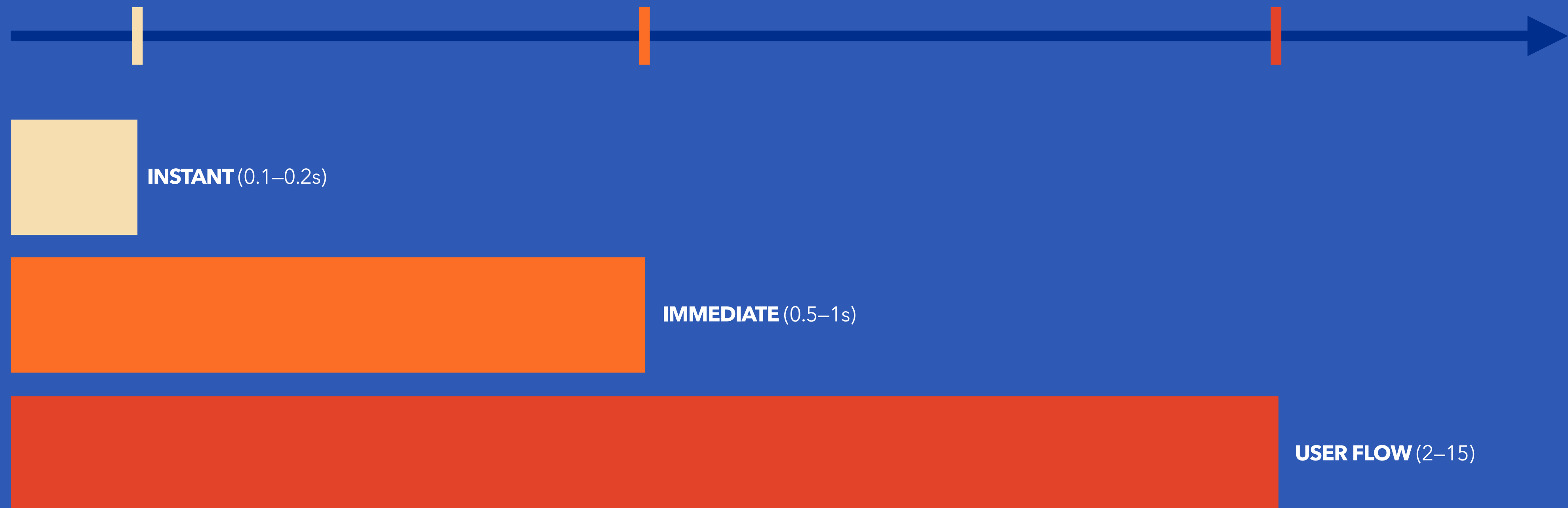
**1.6** SECONDS

# OPTION #2



**2.0** SECONDS

# 1. PERFORMANCE BUDGET

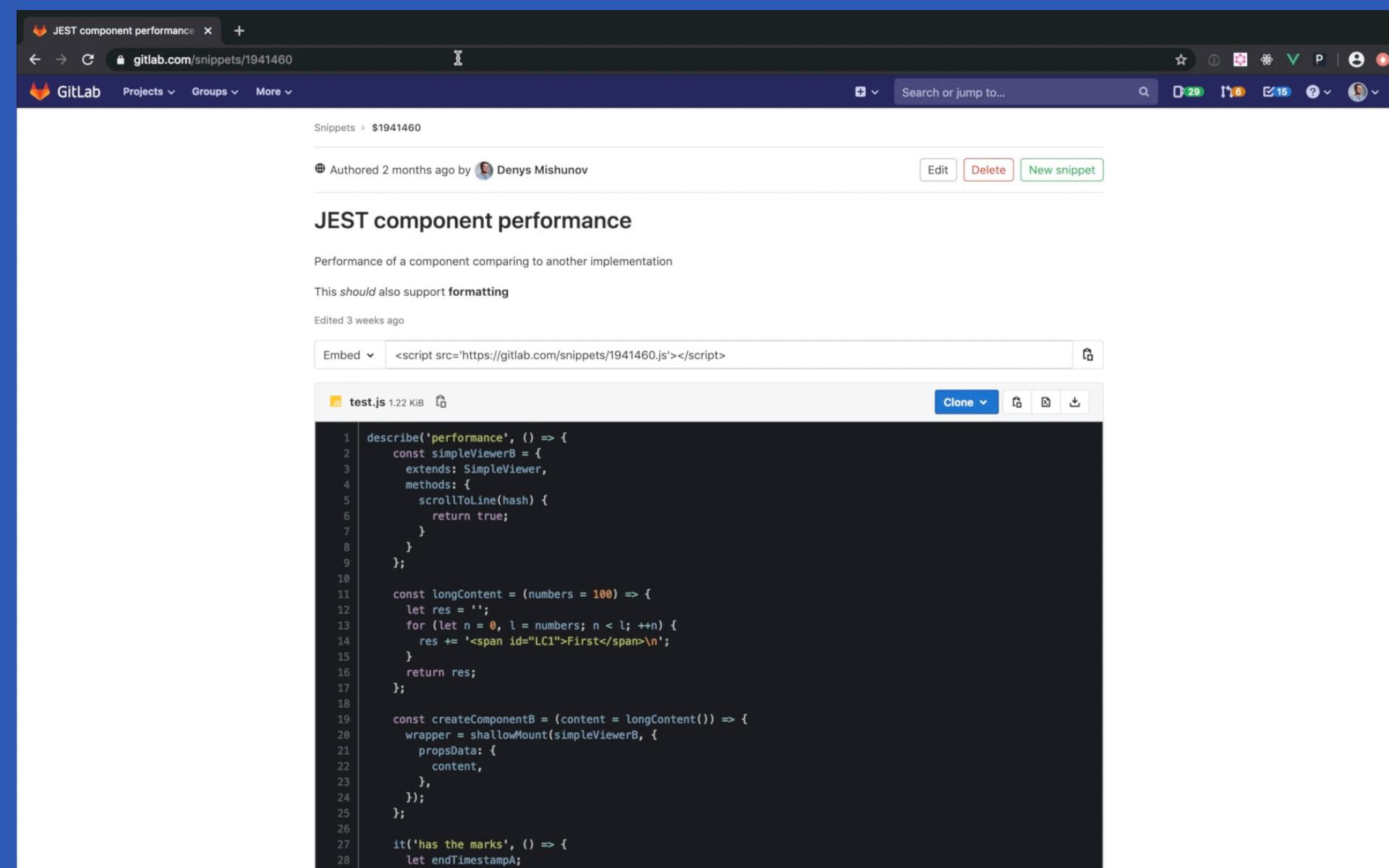


# 2. THE NEED

OPTIMISE TO IMPROVE BAD METRICS

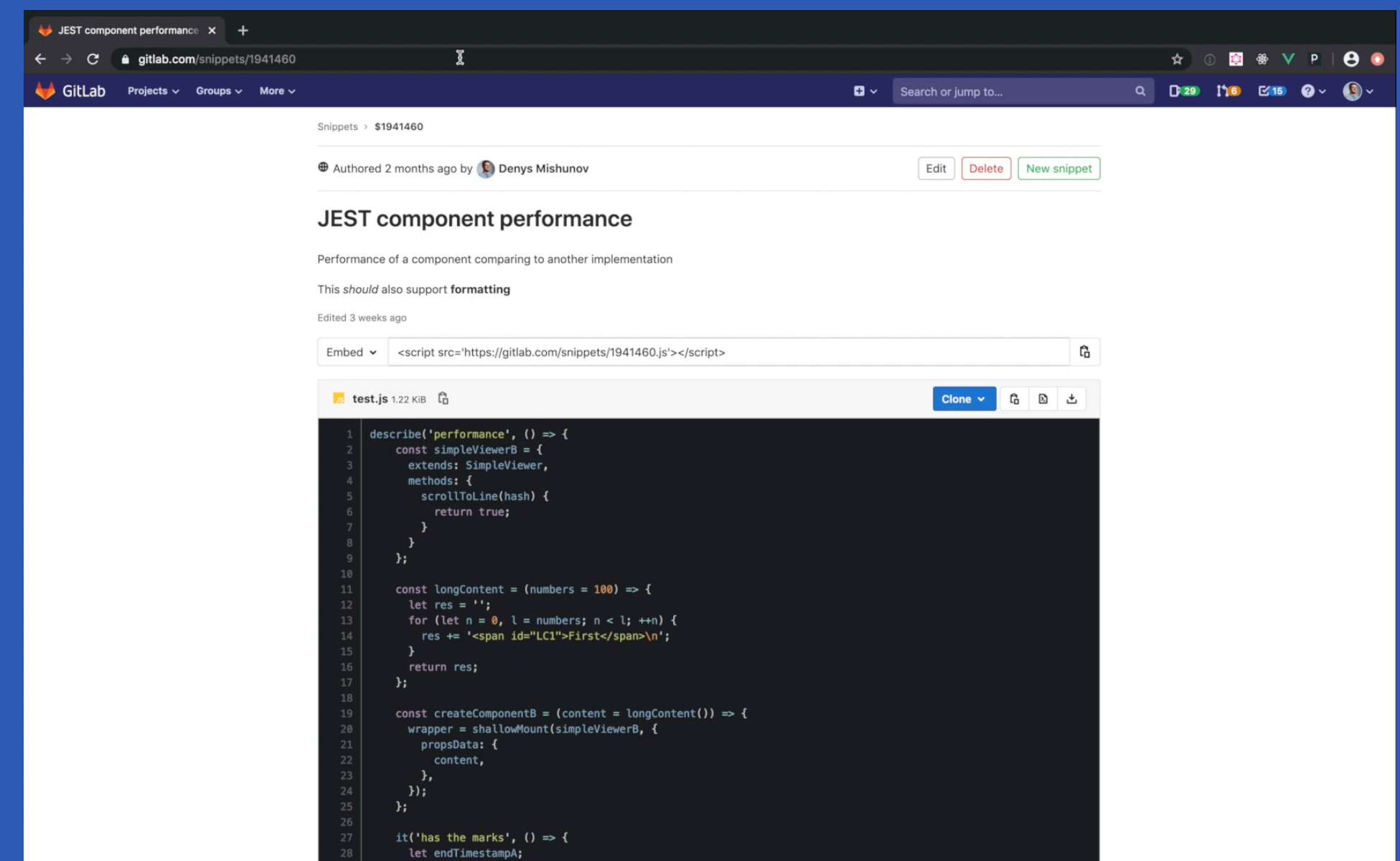


# OPTION #1



1.6 SECONDS

# OPTION #2



2.0 SECONDS

OPTION #1

OPTION #2

400  
milliseconds

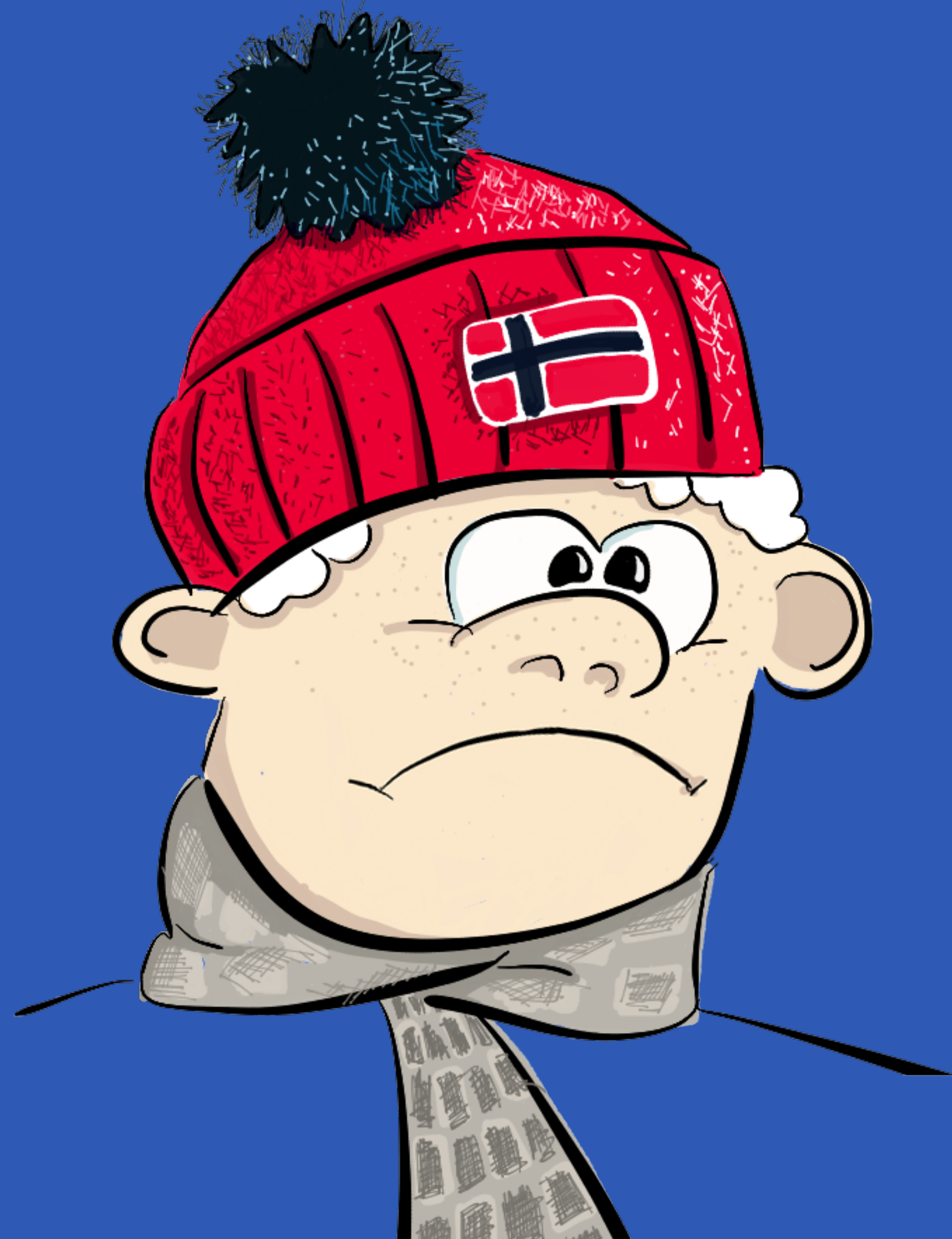


1.6 SECONDS

2.0 SECONDS

# WEBER-FECHNER LAW

JUST NOTICEABLE DIFFERENCE (JND)



EVENT

# WEBER-FECHNER LAW

JUST NOTICEABLE DIFFERENCE (JND)



EVENT

20%

JUST  
**NOTICEABLE**  
DIFFERENCE

NO



LE

# 3. CHASING THE LEADER

OPTIMISE TO SURVIVE COMPETITION



### 3. CHASING THE LEADER

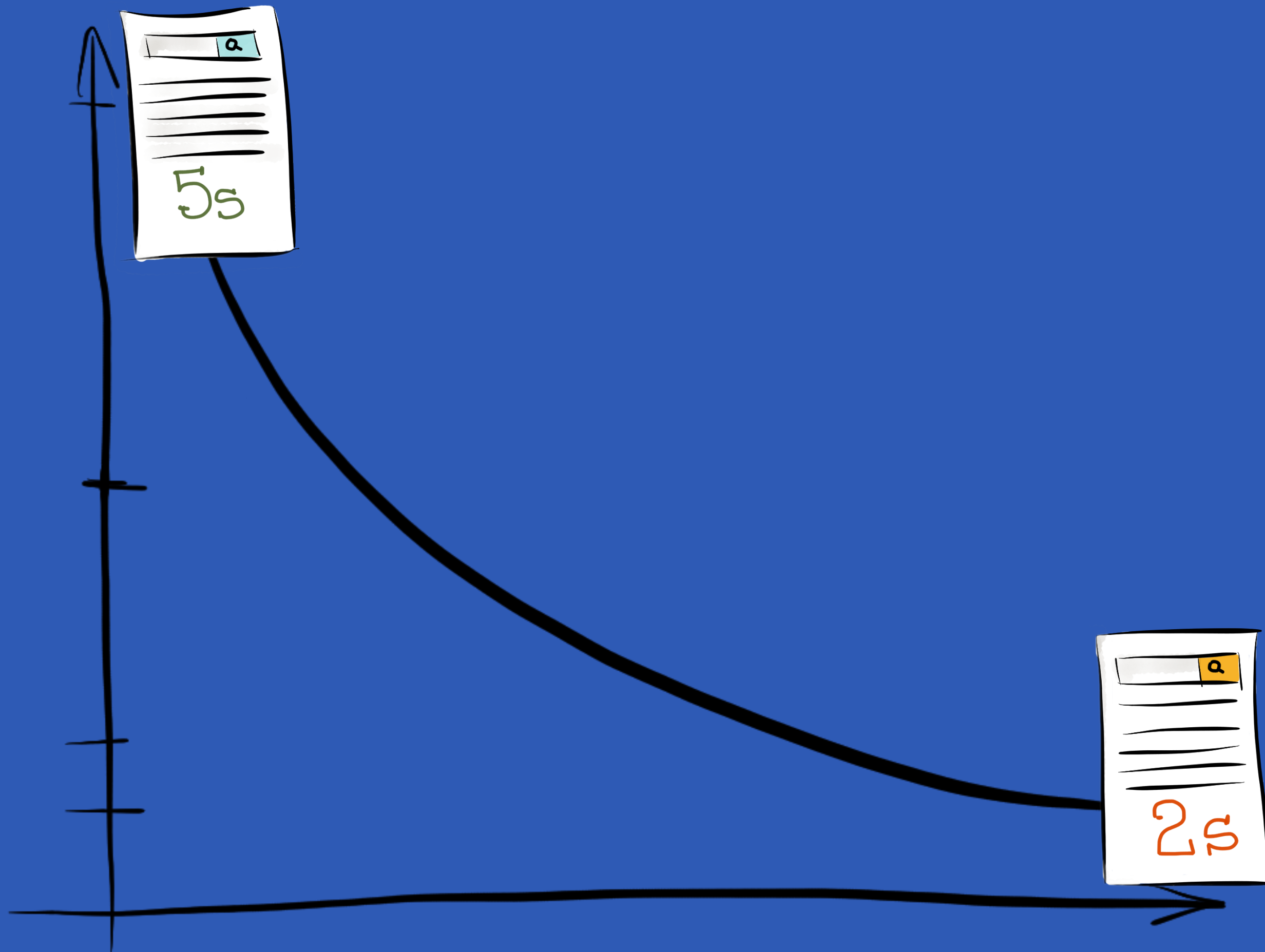
YOU



YOU WISH

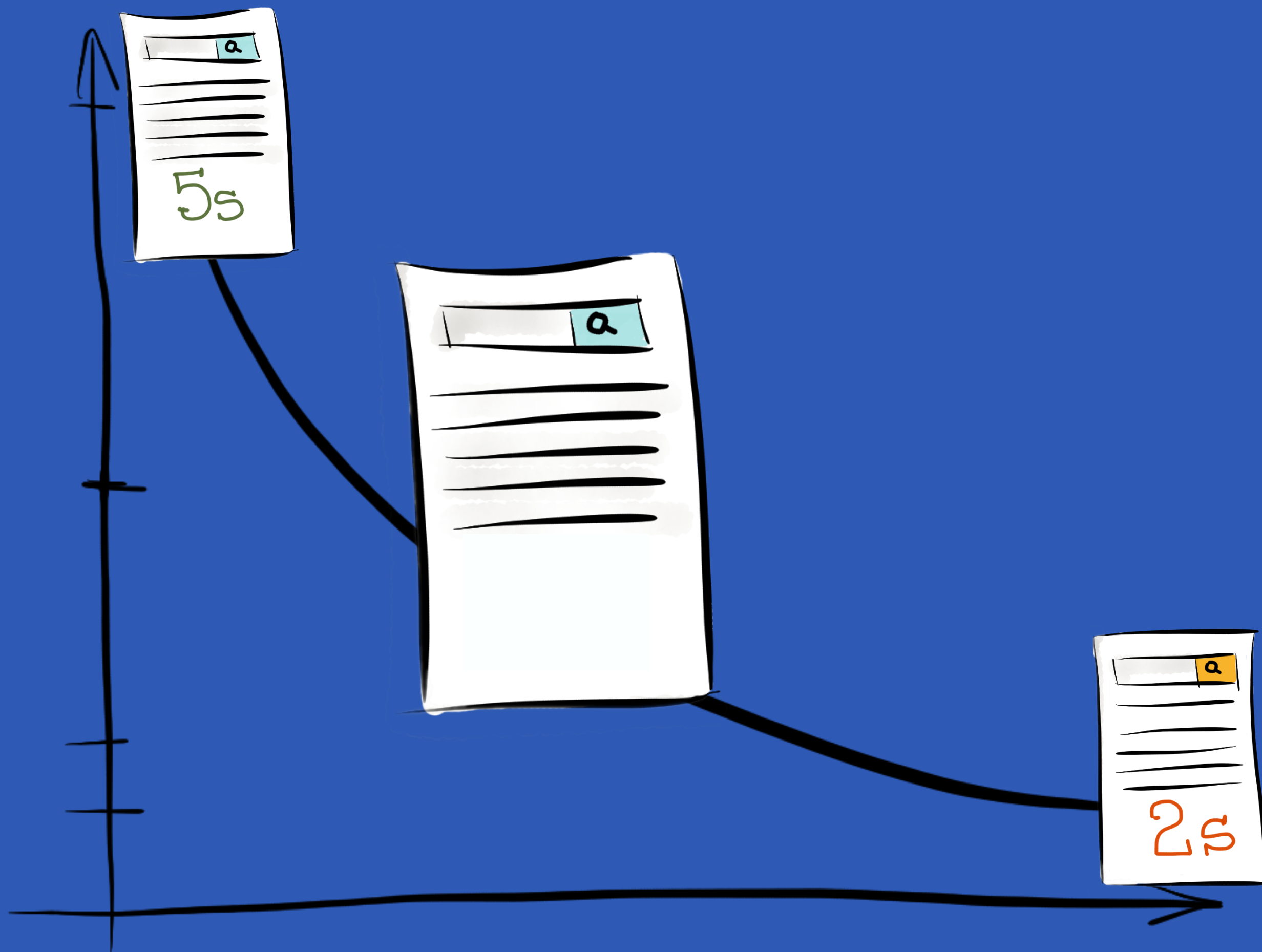
# 3. CHASING THE LEADER

## GEOMETRIC BISECTION



# 3. CHASING THE LEADER

## GEOMETRIC BISECTION



# 3. CHASING THE LEADER

## GEOMETRIC BISECTION



# 3. CHASING THE LEADER

## GEOMETRIC BISECTION



# **IS IT NEEDED?**

- 1. PERFORMANCE BUDGET**
- 2. THE NEED**
- 3. CHASING THE LEADER**



---

#thanks

**TIME SPENT EDUCATING ME DURING THE COFFEE-CHATS**  
**VALUABLE COMMENTS AND HELP**

RAMYA AUTHAPPAN • ALEX BUIJS • JEFF CROW  
JEROME NG • JEREMY JACKSON • GEORGI N. GEORGIEV  
PEDRO POMBEIRO • VIJAY HAWOLDAR • ASH MCKENZIE