



Web Frontend
**Rendering
Performance**
Knowledge & Tips

HTML5 Conference
CyberAgent, Inc. Ayumu Sato

Ayumu Sato

@ahomu

株式会社サイバーエージェント

フロントエンドエンジニア

<http://aho.mu>



さとうあゆむ

@あほむ

縁のキャラくない渋谷サラリーマン

Excel 職人だったりフロント開発だったり

<http://aho.mu> ← 続きはWebで



10 years ago…

愛すべきWeb

市外局番 0480



Aiseikai Hospital
(医) 愛生会病院

05567193



346-0003
埼玉県久喜市
中央 4-3-1

「一文字」が動かず、BGMが鳴らない様です！一部では画面の基調である表構造が消えないかも



愛生会

診療時間
母乳学級へのお説い
産婦人科担当医

What place ?

Information
Gallery

Recruit Recruit



Welcome !

心から歓迎！
www.aiseikai.or.jp

看護部スタッフ 常勤医 Profile
監理部スタッフ para-med.Staff

●特別企画：子宮内の赤ちゃんの心臓の鼓動音を聞いてみませんか？●

文字サイズは「中」で！

GM-Swic



「当院名称」由来の頃

左：一壁口ビー

駅 運



rodinný PENZION 700



na Hrubých lukách



lukách

naše specializace: DOVOLENÁ PRO RODINY S DĚTMI

telefon **607 700 200**

+420607700200

50°25'46.322"N, 15°50'14.309"E

Email

penzion@hrubasy.cz

144454



pokoje



pro děti



fauna



rodiče



tipy na výlet



mapa



Last Minute



★ HOST ★

★ HOST ★

vasion fuer Familien mit Kindern



Kinder- und Familienfreundlich



Ubytování Dvůr Králové nad Labem, Hradecká 497

* KDO SI HRAJE, NEZLOBÍ! *

Check in 15 - 18 hodin (nebo dohodou!)

fris

DHTML

HTML5

Animated GIF

<canvas>

<marquee>

transform
@keyframes

SEARCH

Search by destination

Check-In Check-out

13 11/11/2013

[SEARCH](#)[400 HOTELS](#)[GERMANY](#)[SPAIN](#)[NETHERLANDS](#)[ITALY](#)[SICILY](#)[EASTERN EUROPE](#)[MORE EUROPE](#)

400 HOTELS
WITH SOMETHING
FOR EVERYONE

400 WAYS to BEAT THE WINTER



Magical winter experiences with
NH HOTELS



EUROPEAN
GETAWAYS
FOR EVERY
BUDGET

[SCROLL DOWN](#)
to beat the winter



結局

関心を引くWebサイトは重くなりがち

レンダリングとか

スクロールとか

アニメーションとか

とにかく色々



ms

milli second





“陸上競技や競泳でもタイムの計測は 100分の1秒まで。
しかし、冬季種目のリュージュ(そり)や自動車レースの
F1 などは 1000分の 1秒まで計測している。”

– ミリ秒 - 用語解説辞典 | NTTPCコミュニケーションズ

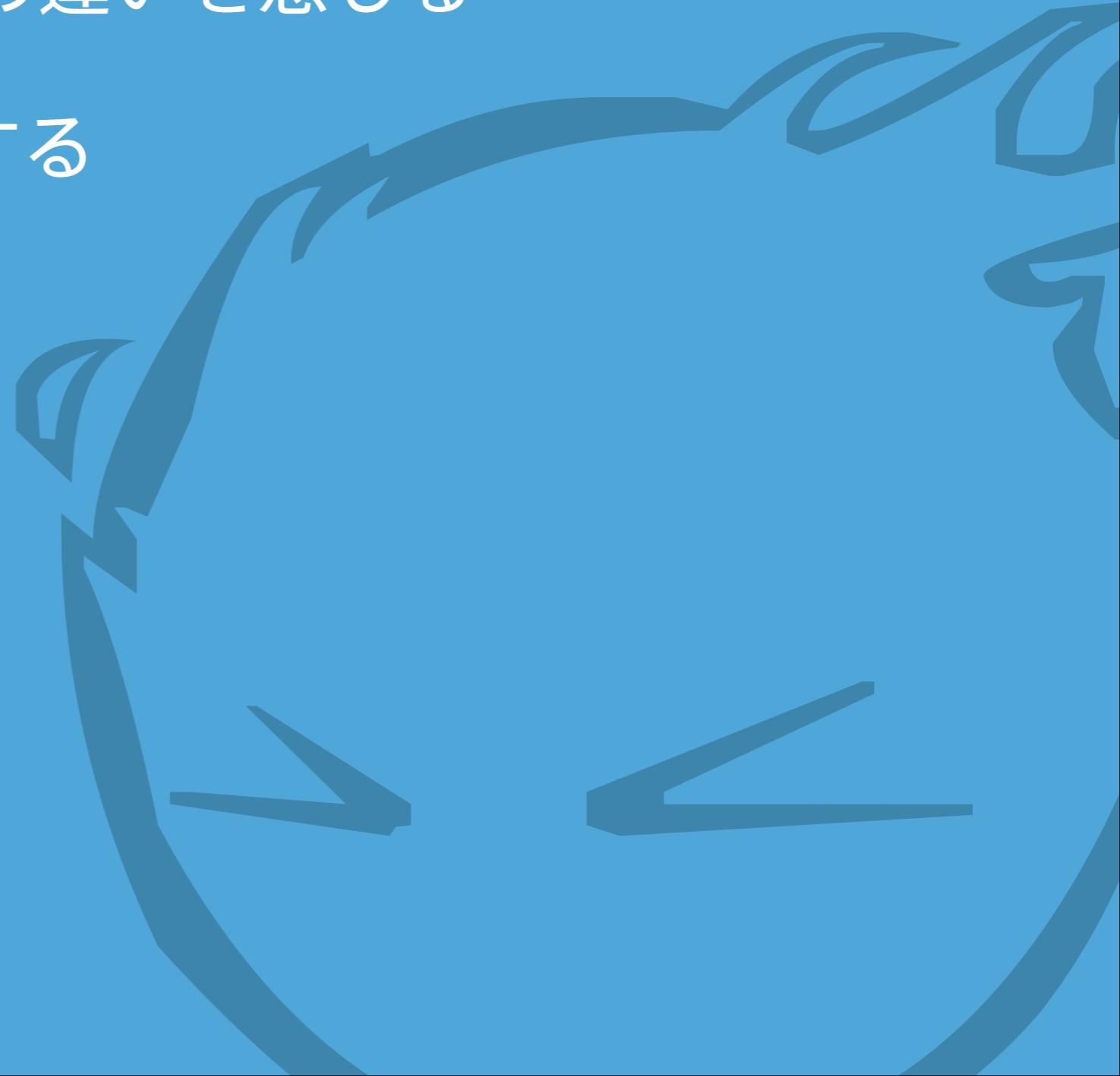
msの努力

人間は50msと100msの違いを感じる

1msからコントロールする

ms単位の工夫が

スムーズな動きを作る

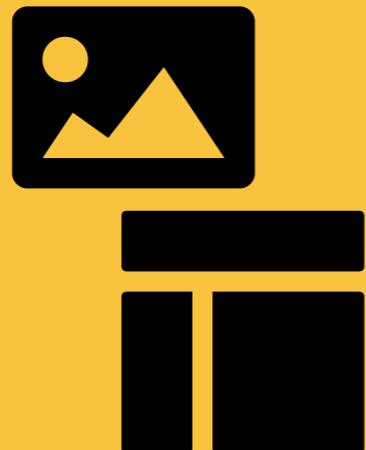


Initial

Load



Display

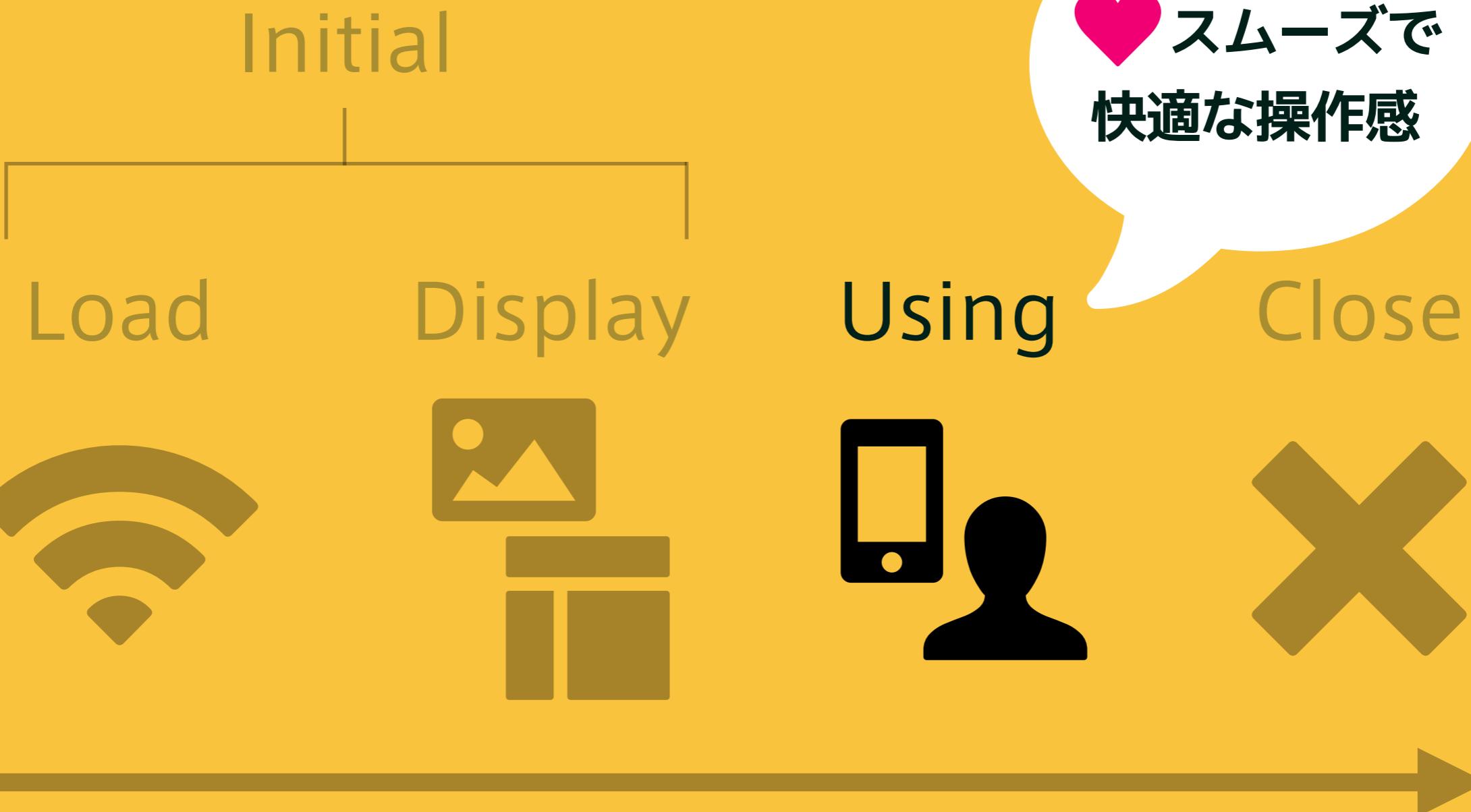


Using



Close





Agenda

1. Setup
2. FPS & Timeline
3. Browser Rendering
4. Case Study
5. Conclusion

前置き

ブラウザのきもちになって考えるために

WebKit 前提気味 (が、原理は大差ない)

Canvasのゴリゴリした話はない

WebGLとかの話もない

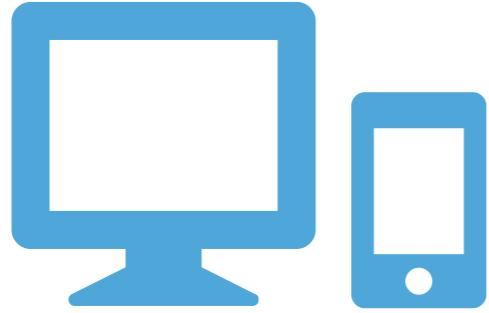
海外のGooglerに感謝を込めて

FPS & Timeline

FPSのはなし

60FPS ?

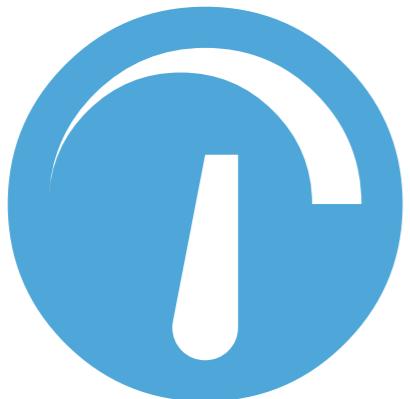
リフレッシュレートとFPS



一般的なディスプレイは
毎秒60回 表示を更新する (60hz)



映画・アニメは 24FPS
30FPS 付近でスムーズっぽい



60FPS にもなるといわゆる
ぬるぬる動くゲームのイメージ

16.666...

$1000\text{ms}/60\text{f} = 16.7\text{ms}/1\text{f}$

16.7ms?



Event Layout



GC

Script

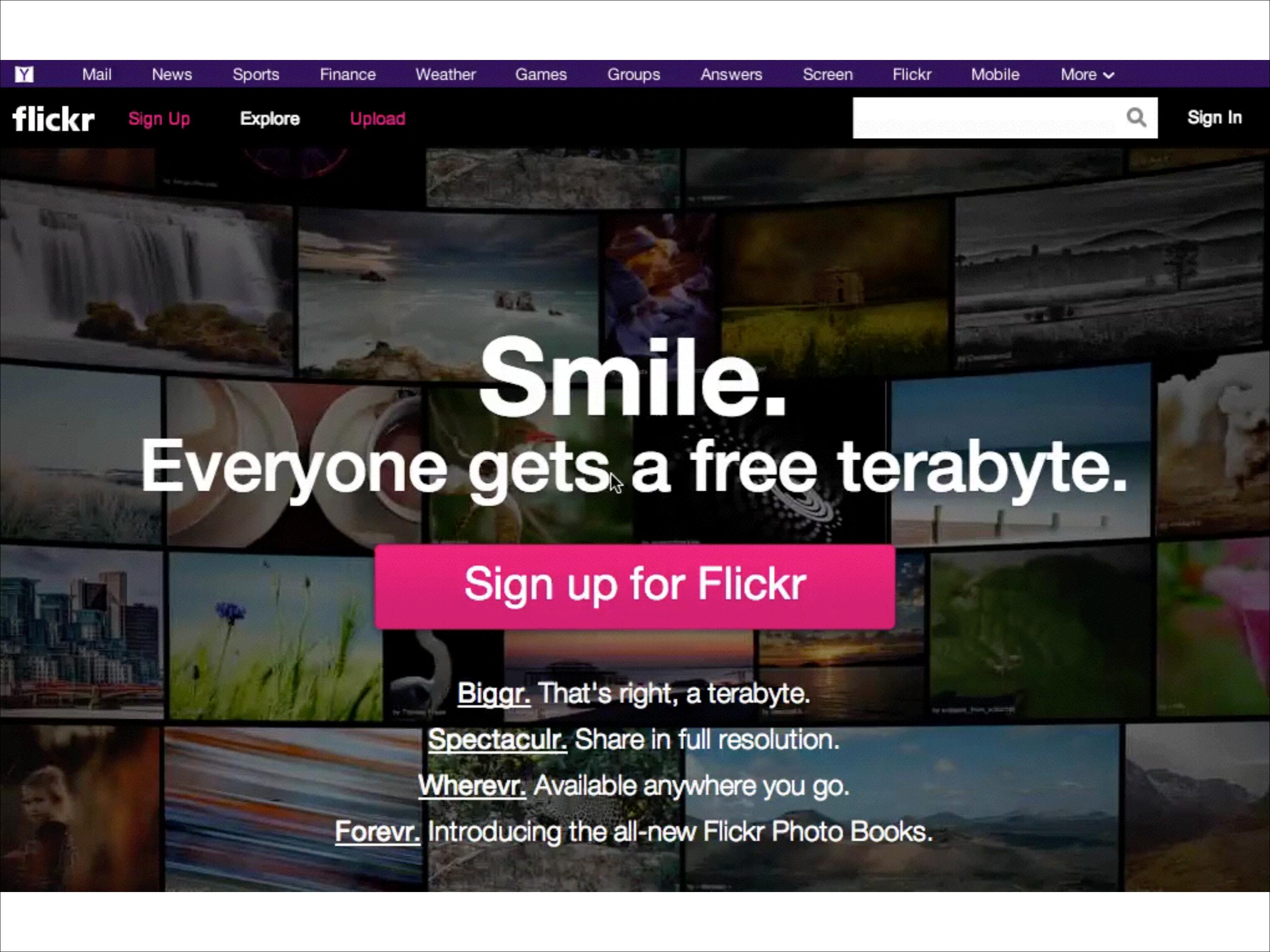
Paint

Developer Tool > Timeline

ChromeならCanaryがオススメ
最新の機能がイチ早く使える

動かない日があっても泣かない（重要）





Smile.
Everyone gets a free terabyte.

Sign up for Flickr

Biggr. That's right, a terabyte.

Spectaculr. Share in full resolution.

Wherevr. Available anywhere you go.

Forevr. Introducing the all-new Flickr Photo Books.

Elements Resources Network Sources Timeline Profiles Audits Console

»

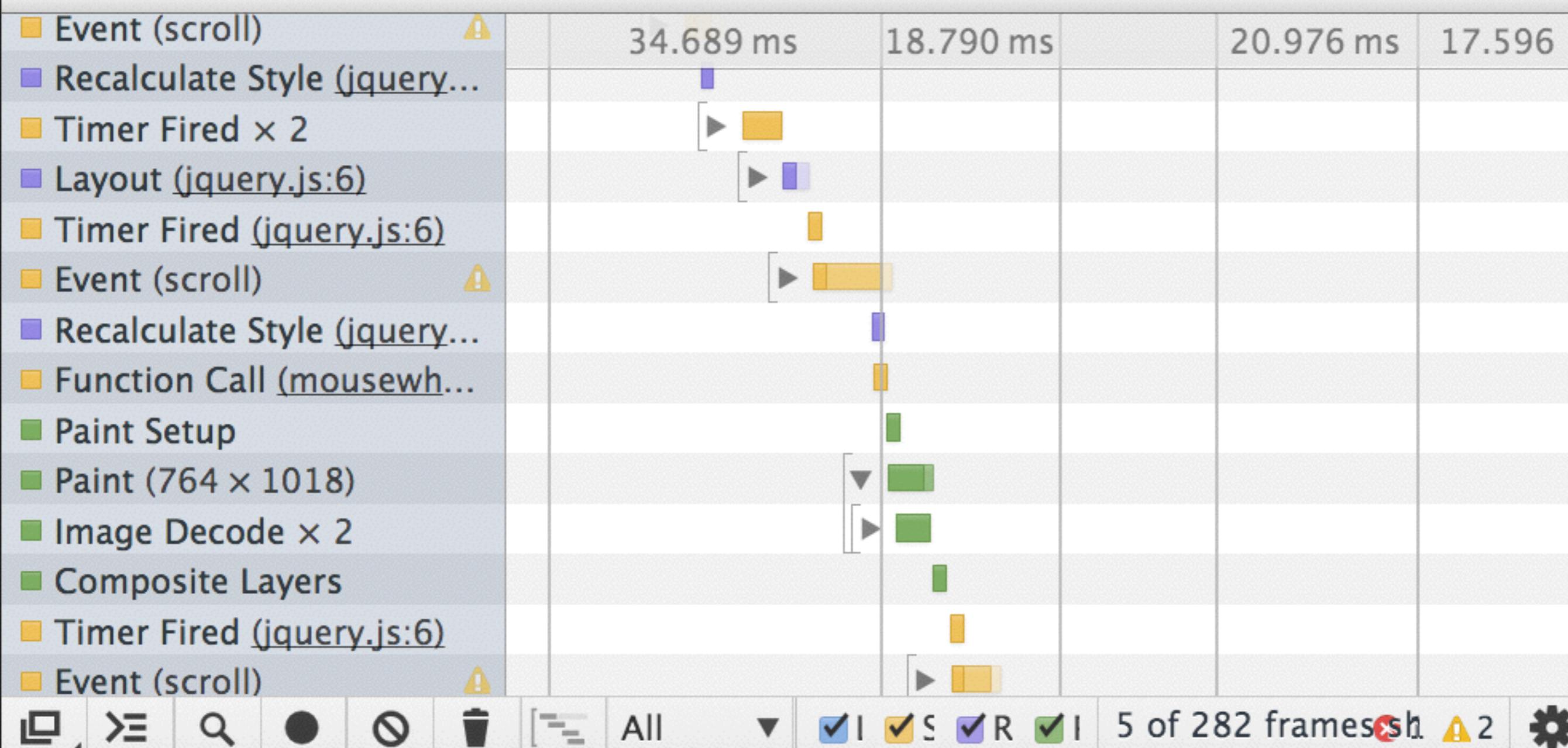
Events

Frames

Memory

30 fps

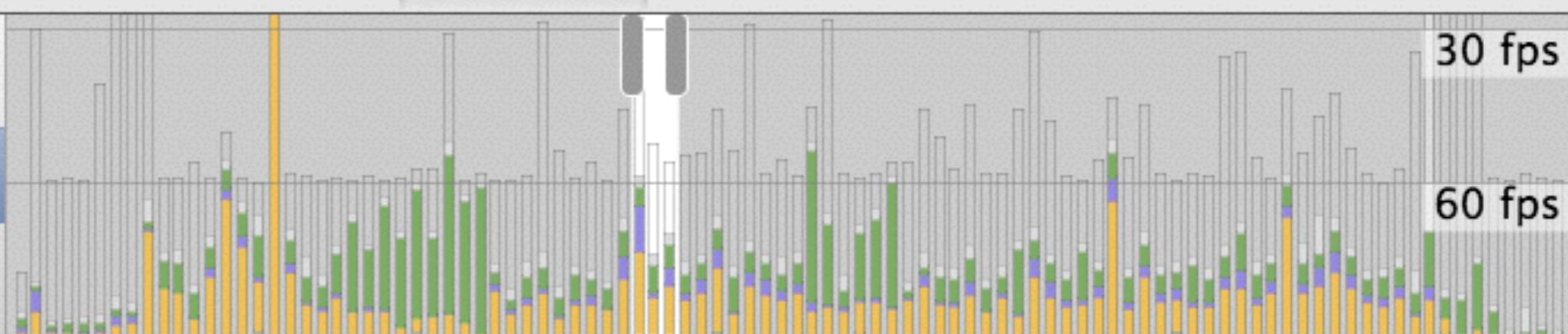
60 fps



Events

Frames

| ~ Memory



- Frame - Details**

Duration 34.689 ms (at 2.26 s)
FPS 28
CPU time 18.292 ms
Aggregated Time 1.477 ms 2.399 ms 5.225 ms 9.191 ms

Timeline details:

 - Event (scroll) (Yellow)
 - Recalculate Style (jquery.js:6) (Purple) - Duration: 34.689 ms, CPU time: 18.292 ms, Aggregated Time: 5.225 ms
 - Timer Fired x 2 (Yellow)
 - Layout (jquery.js:6) (Purple)
 - Timer Fired (jquery.js:6) (Yellow)
 - Event (scroll) (Yellow)
 - Recalculate Style (jquery.js:6) (Purple)
 - Function Call (mousewheel) (Yellow)
 - Paint Setup (Green)
 - Paint (764 x 1018) (Green)
 - Image Decode x 2 (Green)
 - Composite Layers (Green)
 - Timer Fired (jquery.js:6) (Yellow)
 - Event (scroll) (Yellow)

Frame - Details

Duration 34.689 ms (at 2.26 s)

FPS 28

CPU time 18.292 ms

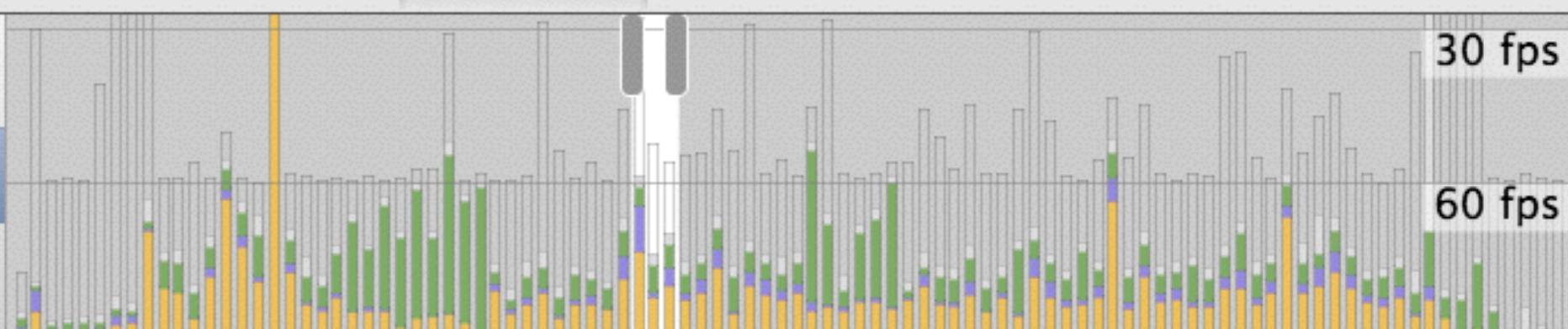
Aggregated Time █ 1.477 ms █ 2.399 ms █ 5.225 ms █ 9.191 ms



Events

Frames

| ~ Memory



- The screenshot shows the Chrome DevTools Timeline tab. The timeline displays several events:

 - Event (scroll)**: A yellow event at the start of the timeline.
 - Recalculate Style (jquery...)**: A purple event starting at 34.689 ms.
 - Timer Fired × 2**: Two orange events starting at 34.689 ms and 18.790 ms.
 - Layout (jquery.js:6)**: A purple event starting at 20.976 ms.
 - Timer Fired (jquery.js:6)**: An orange event starting at 17.596 ms.

Each event is accompanied by its duration in milliseconds.

Layout - Details

Duration 0.338 ms (at 2.28 s)

Self Time 0.267 ms

CPU Time 0.338 ms

Aggregated Time 0.338 ms

Nodes that need layout 8

Layout tree size 489

Layout scope Whole document

Layout root #document

Layout invalidated e.getComputedStyle.Wt @ jquery.js:6
x.extend.cssHooks.opacity.get @ jquery.js:6



Events

Event - Details

Duration 5.499 ms (at 2.29 s)

Self Time 0.041 ms

CPU Time 5.932 ms

Aggregated Time 3.449 ms 2.483 ms

[Details scroll](#)

Used Heap Size 16.1 MB (+48.3 KB)

▶ [Timer Fired \(jquery.js:6\)](#)

▶ [Event \(scroll\)](#) ⚠

▶ [Recalculate Style \(jquery...\)](#)

▶ [Function Call \(mousewh...](#)

▶ [Paint Setup](#)

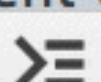
▶ [Paint \(764 × 1018\)](#)

▶ [Image Decode × 2](#)

▶ [Composite Layers](#)

▶ [Timer Fired \(jquery.js:6\)](#)

▶ [Event \(scroll\)](#) ⚠



All

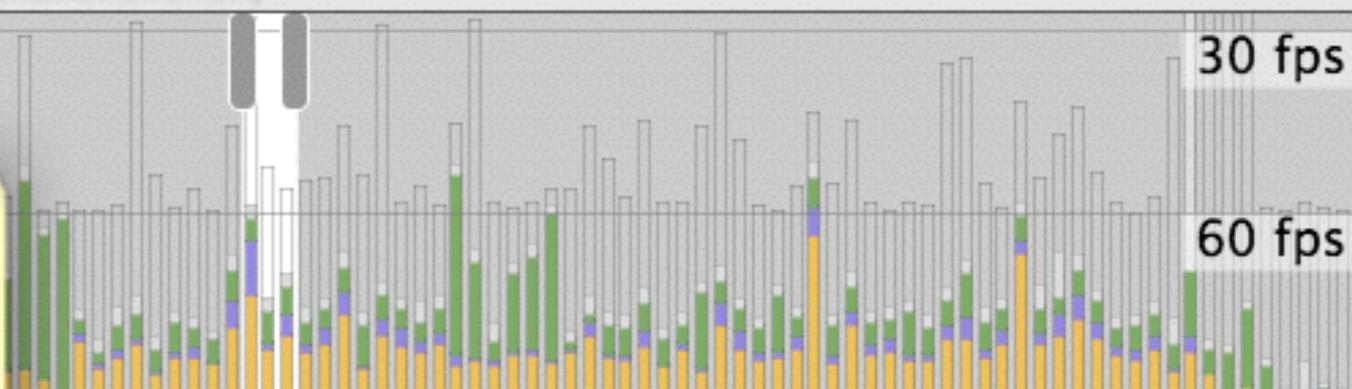


5 of 282 frames ✖ ⚠ 2



30 fps

60 fps



18.790 ms 20.976 ms 17.596

Recalculate Style – Details

Profiles Audits Console

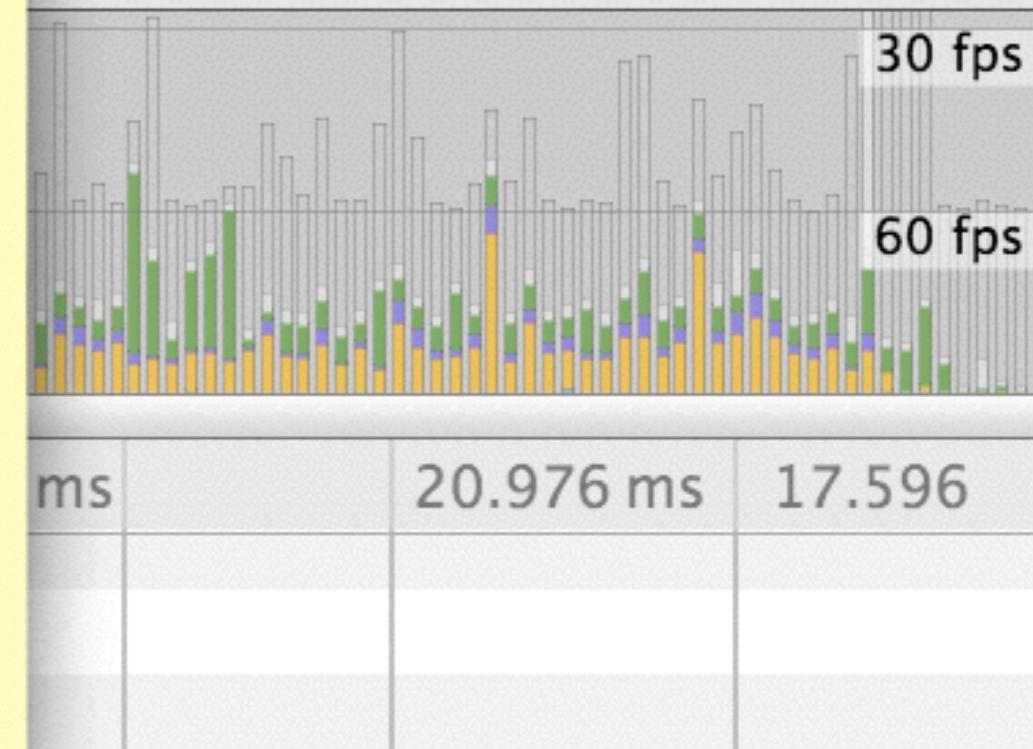
2

Duration 0.115 ms (at 2.29 s)

Elements affected 1

Call Site stack

x.extend.style	@ jquery.js:6
(anonymous function)	@ jquery.js:6
x.extend.access	@ jquery.js:4
x.extend.access	@ jquery.js:4
x.fn.extend.css	@ jquery.js:6
_int._process	@ run.js:657
(anonymous function)	@ run.js:854
x.event.dispatch	@ jquery.js:5
v.handle	@ jquery.js:5



- Event (scroll)
 - Recalculate Style ([jquery...](#))
 - Function Call ([mousewh...](#))
 - Paint Setup
 - Paint (764 × 1018)
 - Image Decode × 2
 - Composite Layers
 - Timer Fired ([jquery.js:6](#))
 - Event (scroll)



Events

Frames

Memory

30 fps

60 fps

Event (scroll)



34.689 ms

Recalculate Style (jquery...)

Function Call - Details

Duration 0.701 ms (at 2.29 s)

Location [mousewheel.js:56](#)

Used Heap Size 16.2 MB (+13.8 KB)

Recalculate Style (jquery...)

Function Call (mousewh...

Paint Setup

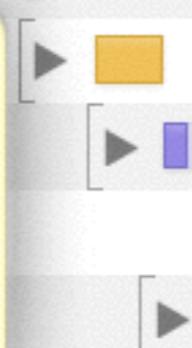
Paint (764 × 1018)

Image Decode × 2

Composite Layers

Timer Fired (jquery.js:6)

Event (scroll)



18.790 ms

20.976 ms

17.596



All



5 of 282 frames

! 2



Events

Frames

Memory

30 fps

60 fps

Paint - Details

Duration 4.732 ms (at 2.29 s)

Self Time 3.890 ms

CPU Time 4.838 ms

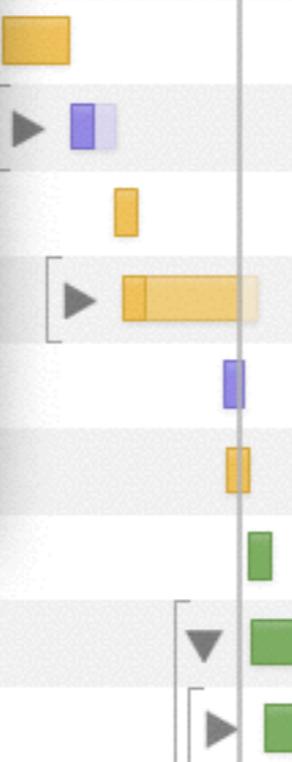
Aggregated Time  4.838 ms

Location (0, -102)

Dimensions 764 × 1018

Layer root #document

9 ms 18.790 ms 20.976 ms 17.596



 Paint setup

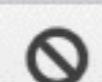
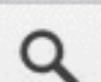
 Paint (764 × 1018)

 Image Decode × 2

 Composite Layers

 Timer Fired ([jquery.js:6](#))

 Event (scroll)



All



5 of 282 frames 

 2



Browser Rendering

レンダリングフローとよくある問題

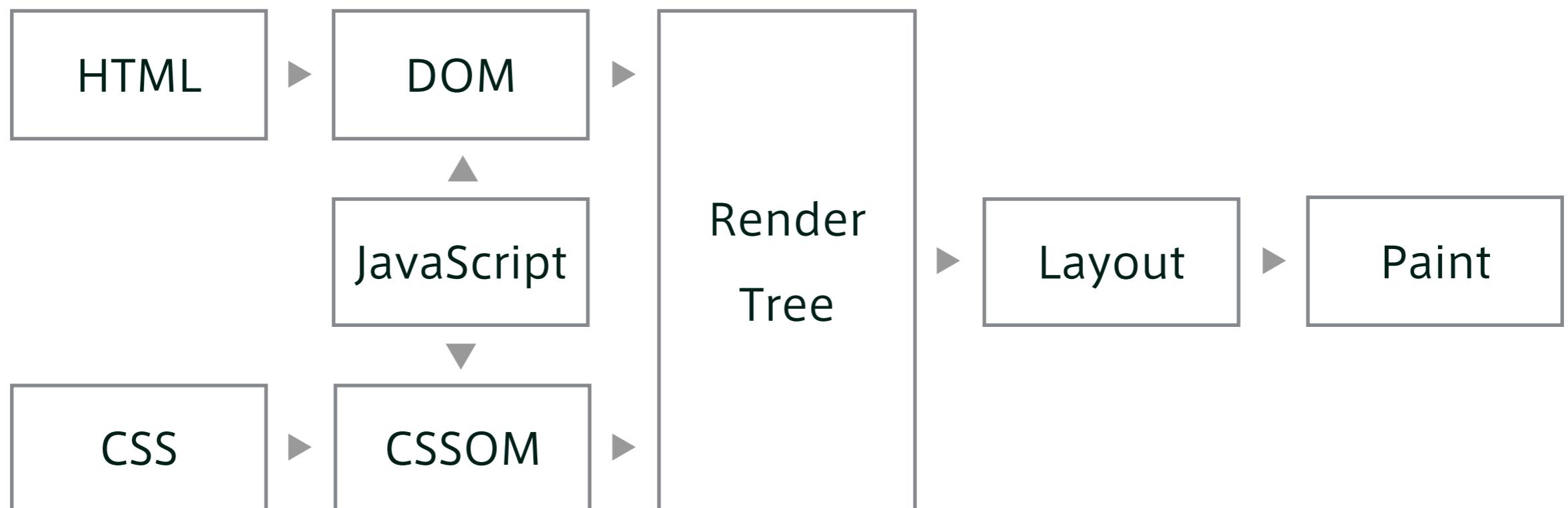
Rendering Engine

今回は WebKit を取り扱います

<http://myakura.github.io/n/webkit4devs.html>

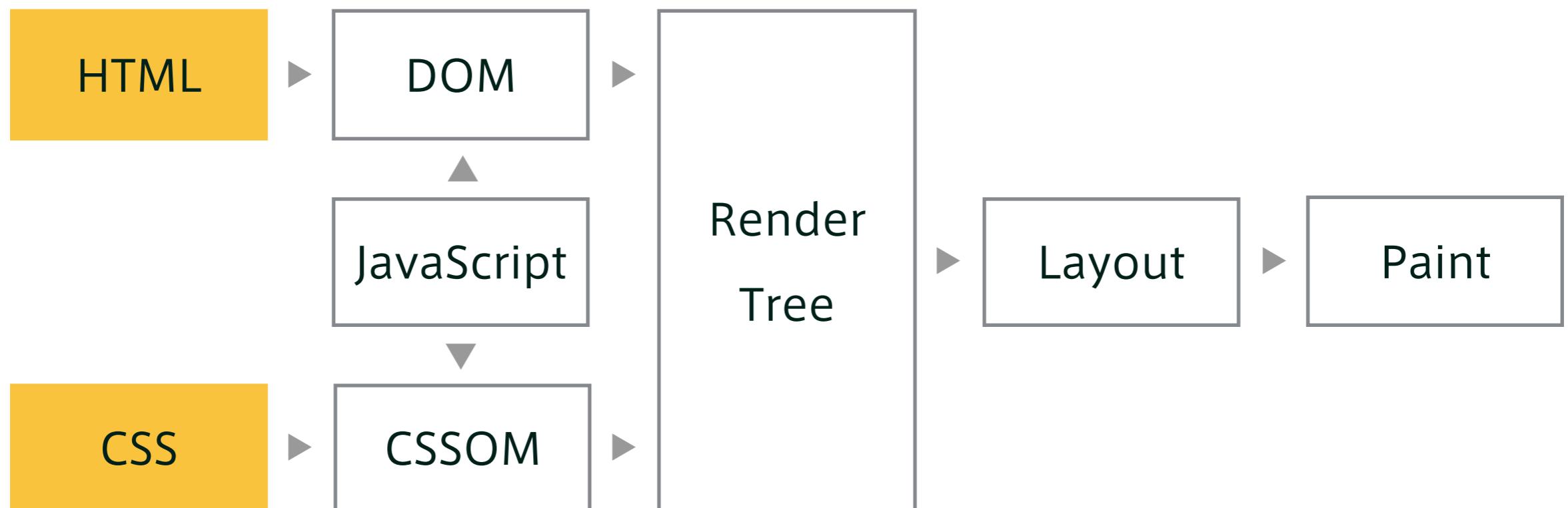
Rendering Flow

レンダリングフロー



Rendering Flow

レンダリングフロー



HTML/CSS

レンダリング処理の元になるもの

The Button



the_button.css

```
/* いわゆるボタン */

.the_button {

    width: 200px;
    height: 50px;

    background: linear-gradient
        (top, #48a2fb, #005cc2);
    border-radius: 5px;
    box-shadow: 0 3px 5px rgba(0,0,0,.6)
}
```

the_button.css

```
.the_button {  
    width: 200px;  
    height: 50px;  
  
    background: linear-gradient  
        (top, #48a2fb, #005cc2);  
    border-radius: 5px;  
    box-shadow: 0 3px 5px rgba(0,0,0,0.6)  
}
```

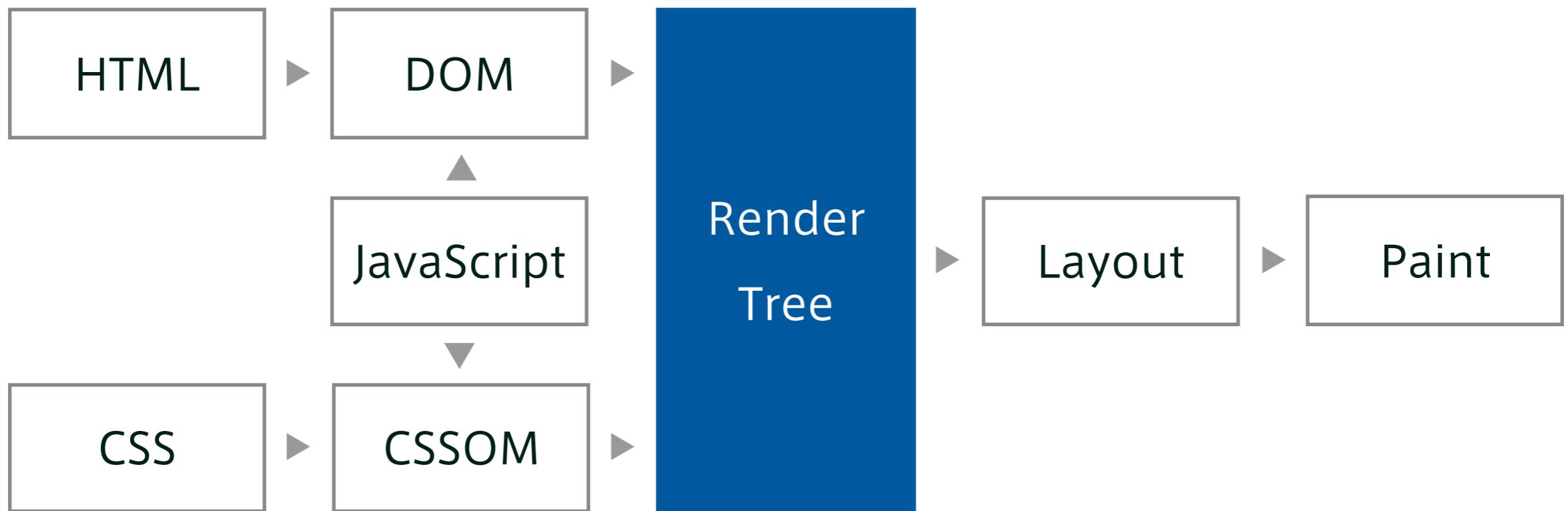
Apply Style

Layout

Paint

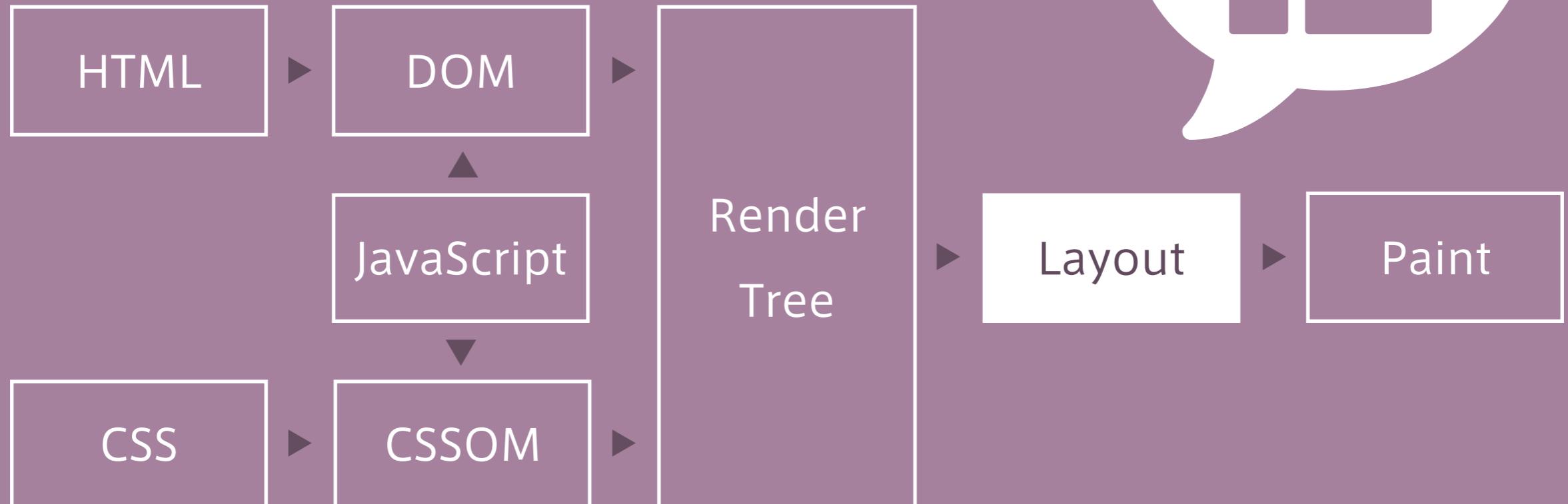
Render Tree

DOMとCSSOMから描画すべき要素をツリー化する



Layout ?

レンダーツリーを元に要素を配置する



<section>

<h2>

<p>



<section>

<h2>

<p>



<section>

<h2>

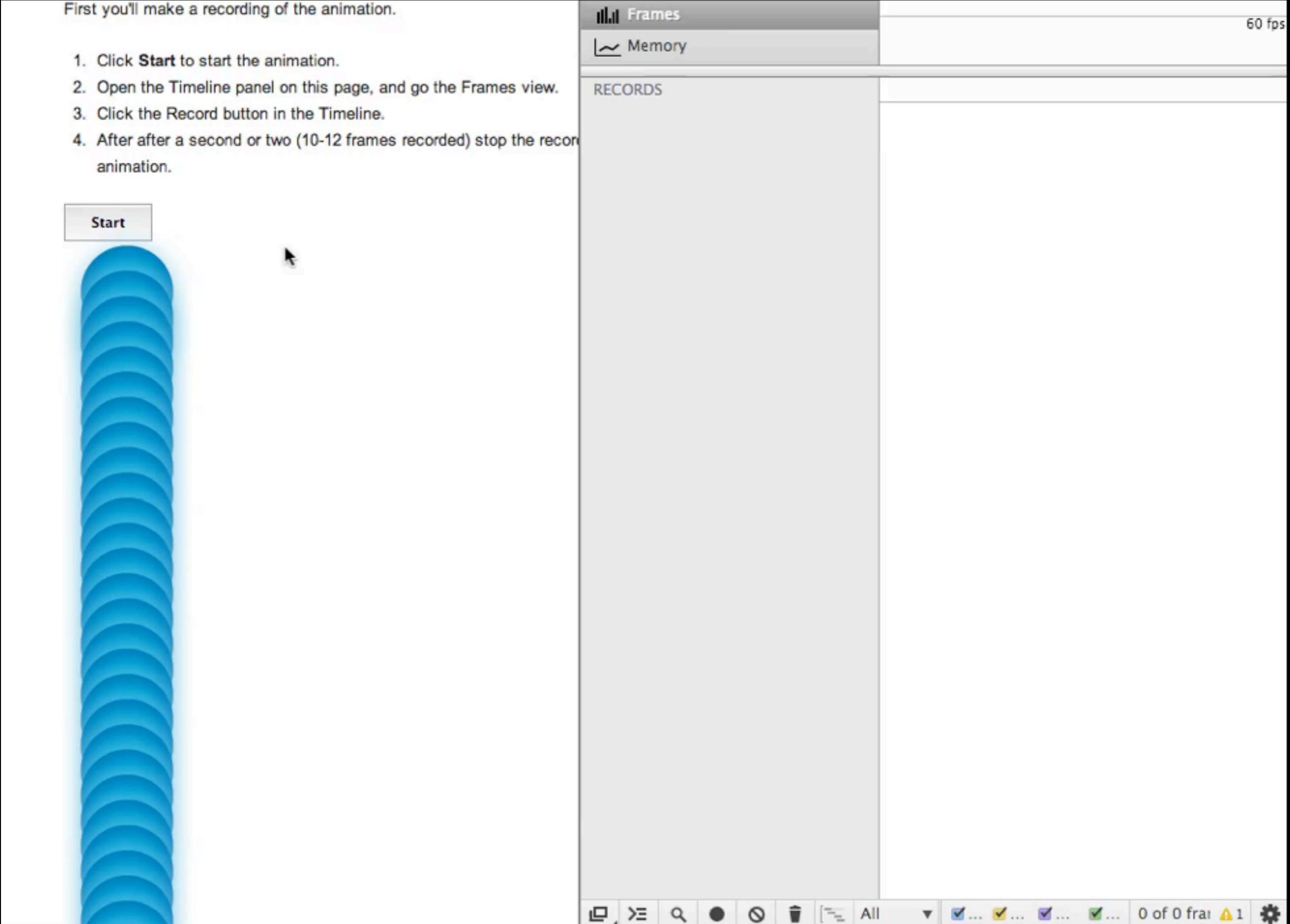
<p>

Timeline > Frames

FPSの推移と共に、フレーム内の処理内訳を知る

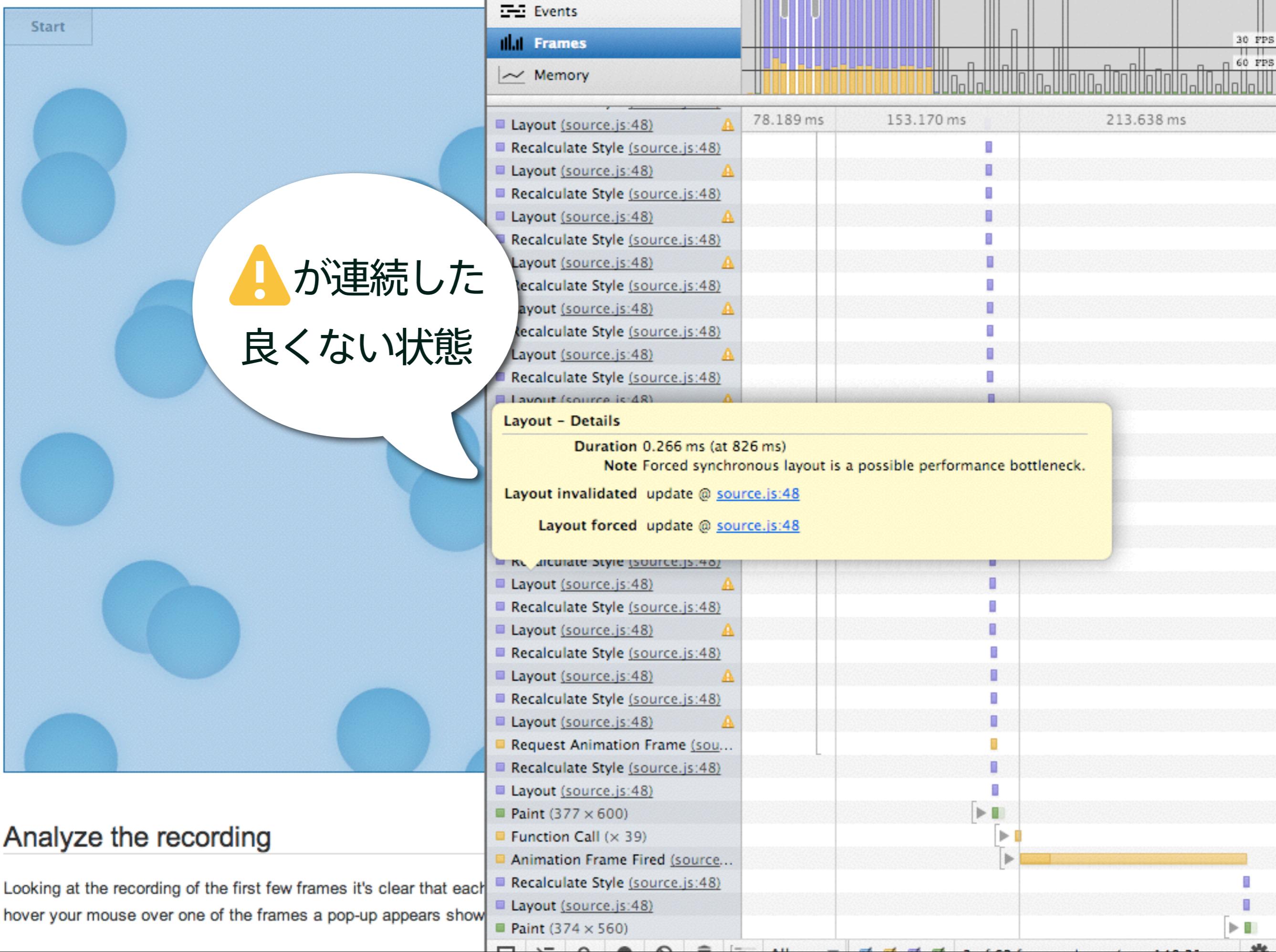
First you'll make a recording of the animation.

1. Click **Start** to start the animation.
2. Open the Timeline panel on this page, and go the Frames view.
3. Click the Record button in the Timeline.
4. After after a second or two (10-12 frames recorded) stop the recording of the animation.



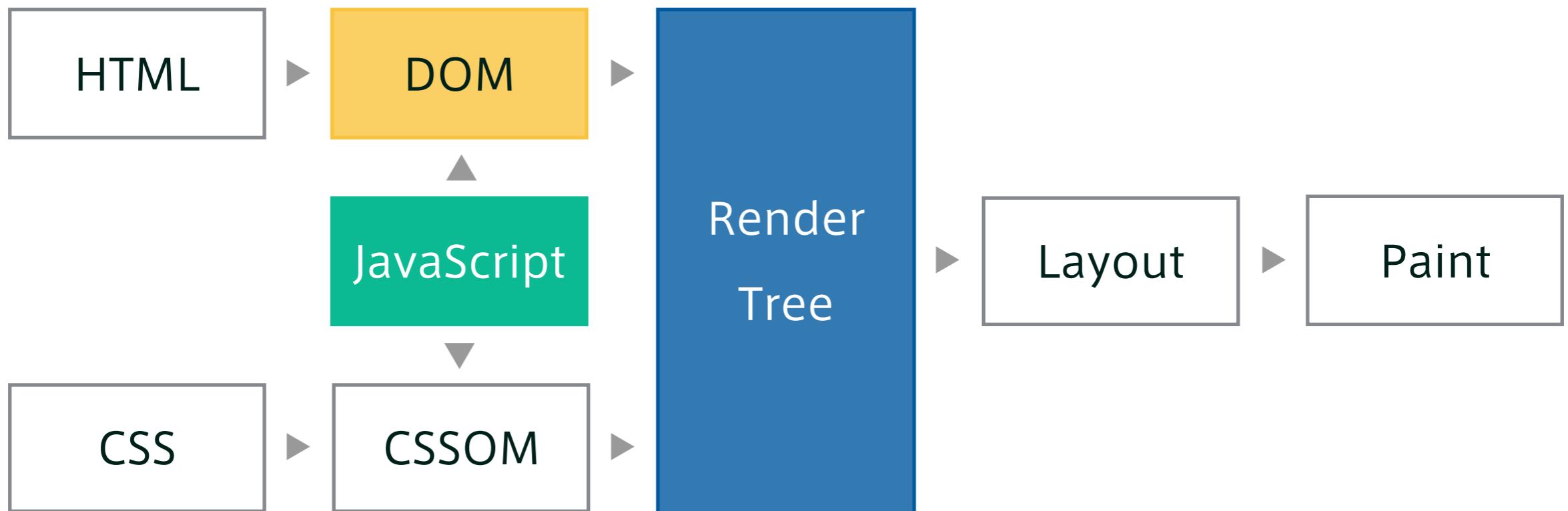
Layout Thrashing

Layout処理の過度な発生



Property Access

スクリプトによるプロパティアクセス



READ > WRITE > READ > WRITE > READ…

```
function update(timestamp) {  
  
    for (var m = 0; m < movers.length; m++) {  
  
        movers[m].style.left = (  
            Math.sin(  
  
                movers[m].offsetTop + timestamp / 1000  
            ) + 1) * 500  
        ) +'px';  
    }  
    raf = window.requestAnimationFrame(update);  
}
```

READ > WRITE > READ > WRITE > READ…

```
function update(timestamp) {  
    var movers = document.querySelectorAll('.mover');  
    for (var m = 0; m < movers.length; m++) {  
        movers[m].style.left = (Math.sin(  
            movers[m].offsetTop + timestamp * 0.001  
        ) + 1) * 500  
        + 'px';  
    }  
    raf = window.requestAnimationFrame(update);  
}
```



READ > WRITE > WRITE > WRITE…

```
function update(timestamp) {  
    var offsetTop = movers[m].offsetTop;  
  
    for (var m = 0; m < movers.length; m++) {  
  
        movers[m].style.left = (  
            Math.sin(  
  
                offsetTop + timestamp / 1000  
            ) + 1) * 500  
        ) +'px';  
    }  
    raf = window.requestAnimationFrame(update);  
}
```

Element

clientHeight clientLeft clientTop clientWidth
offsetHeight offsetLeft offsetTop offsetWidth
scrollHeight scrollLeft scrollTop scrollWidth
innertText outerText getBoundingClientRects
etc...

MouseEvent

layerX layerY offsetX offsetY

Window

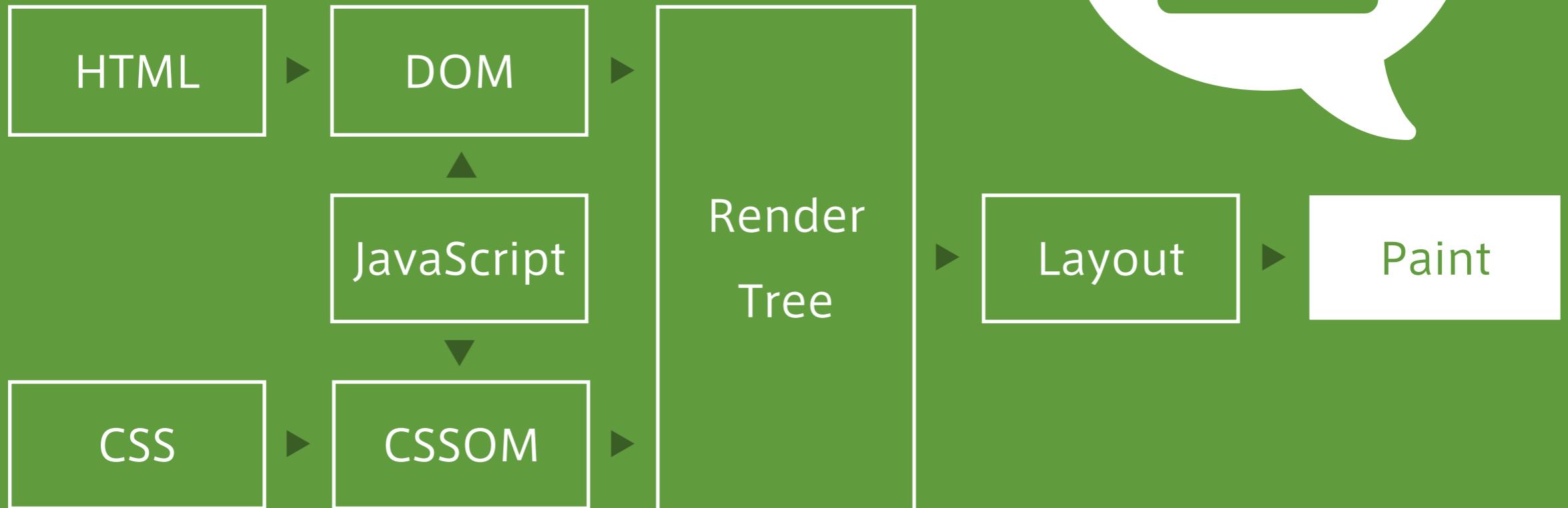
scrollBy scrollTo scrollX scrollY
getComputedStyle

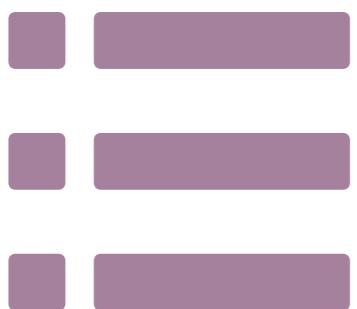
Frame, Document & Image

height width

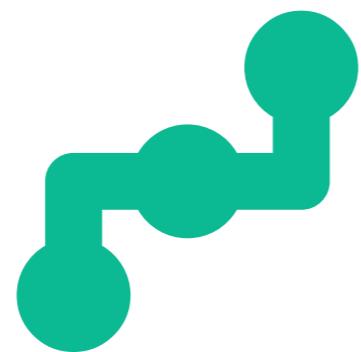
Paint ?

スタイルや画像データを描画する





Styles

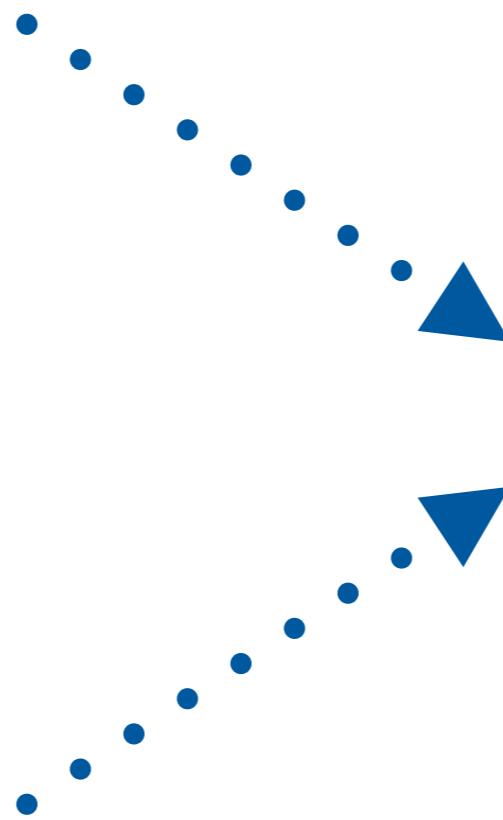


Draw



Image

Rasterize



Decode
(&Resize)



Display

Continuous Page Repainting

画面の描画に何msかかっているのかを知る



We landed on the moon and we left stuff there. A lot of it.

5 9 2 3 7

things left so far.



1 BIG DOUGHNUT



1 LAWNMOWER



1 ASTRO CAT



1 RECLINER



1 MAGIC GNOME



LATEST THING NEWS

New things discovered this week

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure **dolor in reprehenderit** in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate

TOP CATEGORIES

Furniture	185 items	
Fashion	4,729 items	
Pets	4 items	

NEW THING ALERTS

Your Name

Your Email

Too complex Style

負荷の高い、複雑なCSS



(none)
100 %



box-shadow
328 %



border-radius
466%



dotted-border
125 %

[http://havelog.ayumusato.com/develop/performance/e560-
css_rendering_with_skia_debugger.html](http://havelog.ayumusato.com/develop/performance/e560-css_rendering_with_skia_debugger.html)



(none)
100 %



shadow+dotted
403 %



box-shadow
328 %



shadow+radius
2074 %



border-radius
466%



radius+dotted
9892 %



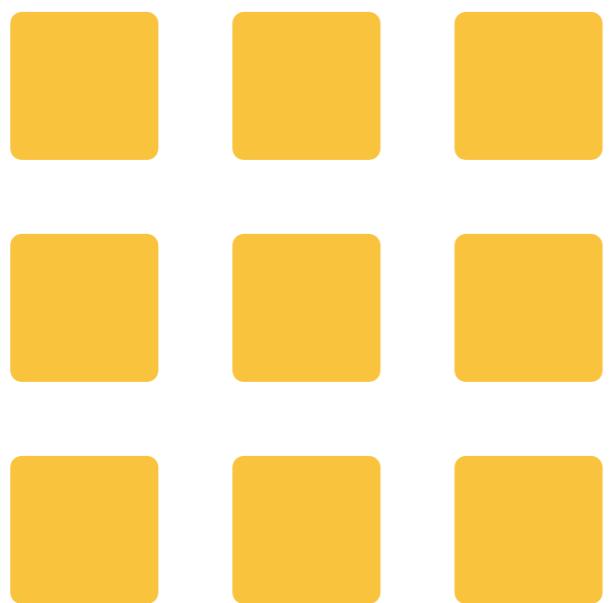
dotted-border
125 %



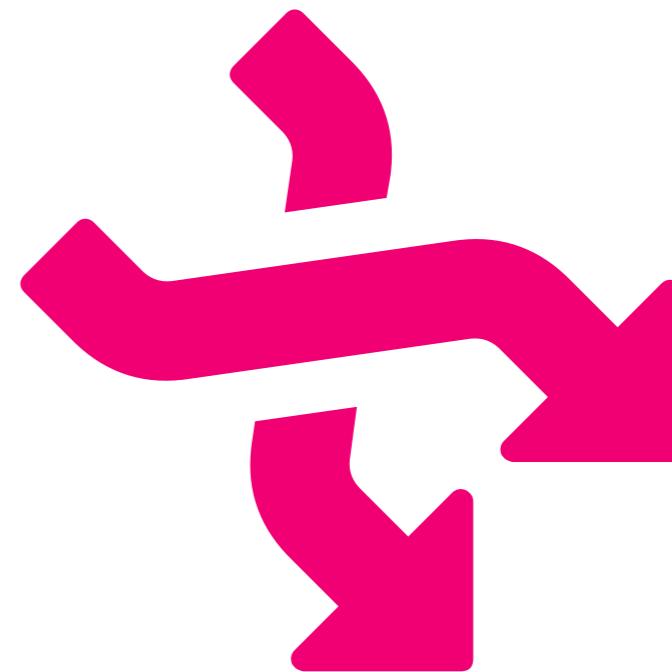
all mix!!
11654 %

[http://havelog.ayumusato.com/develop/performance/e560-
css_rendering_with_skia_debugger.html](http://havelog.ayumusato.com/develop/performance/e560-css_rendering_with_skia_debugger.html)

Area (領域の大きさ)



Complexity (複雑さ)

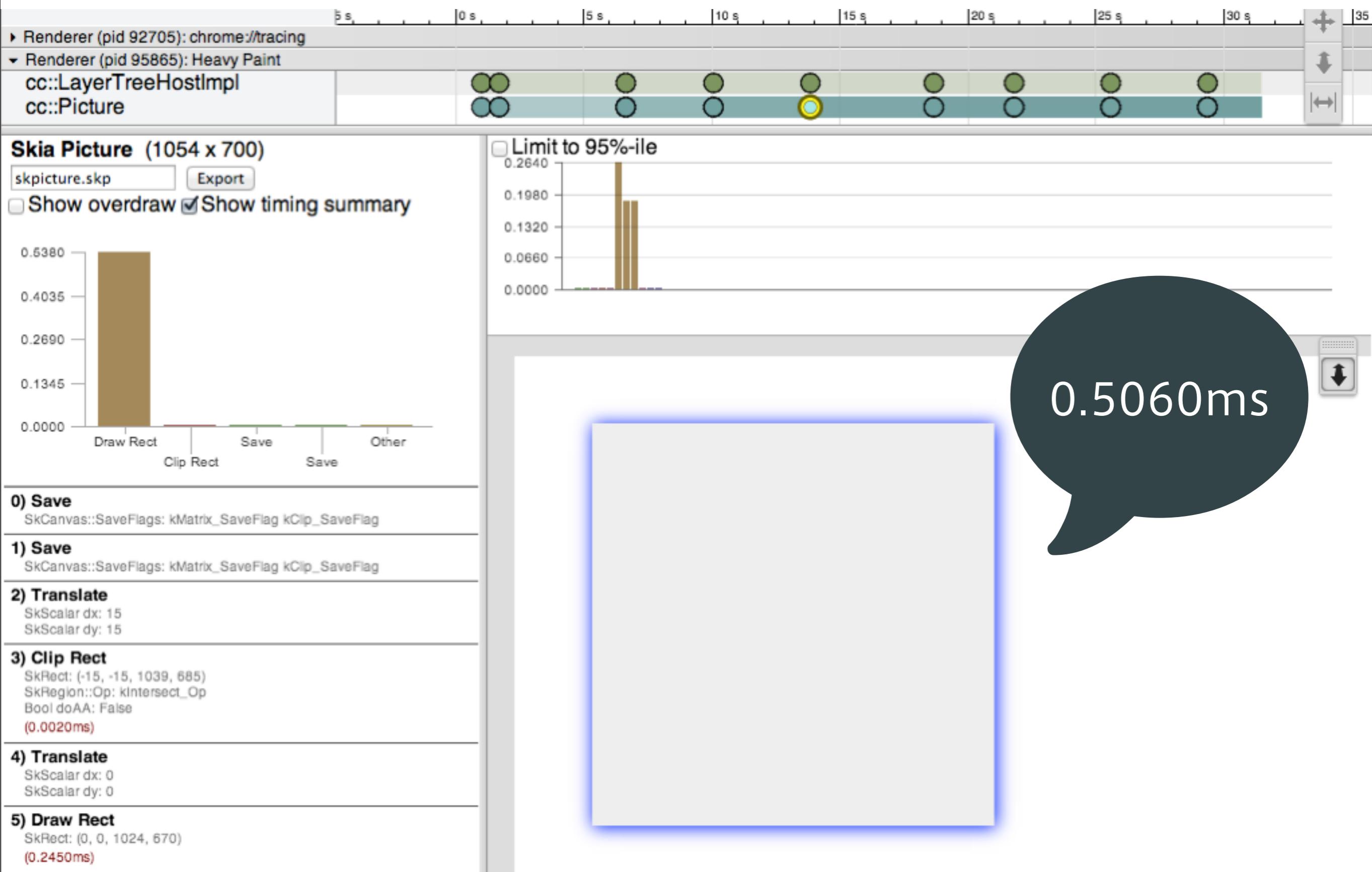


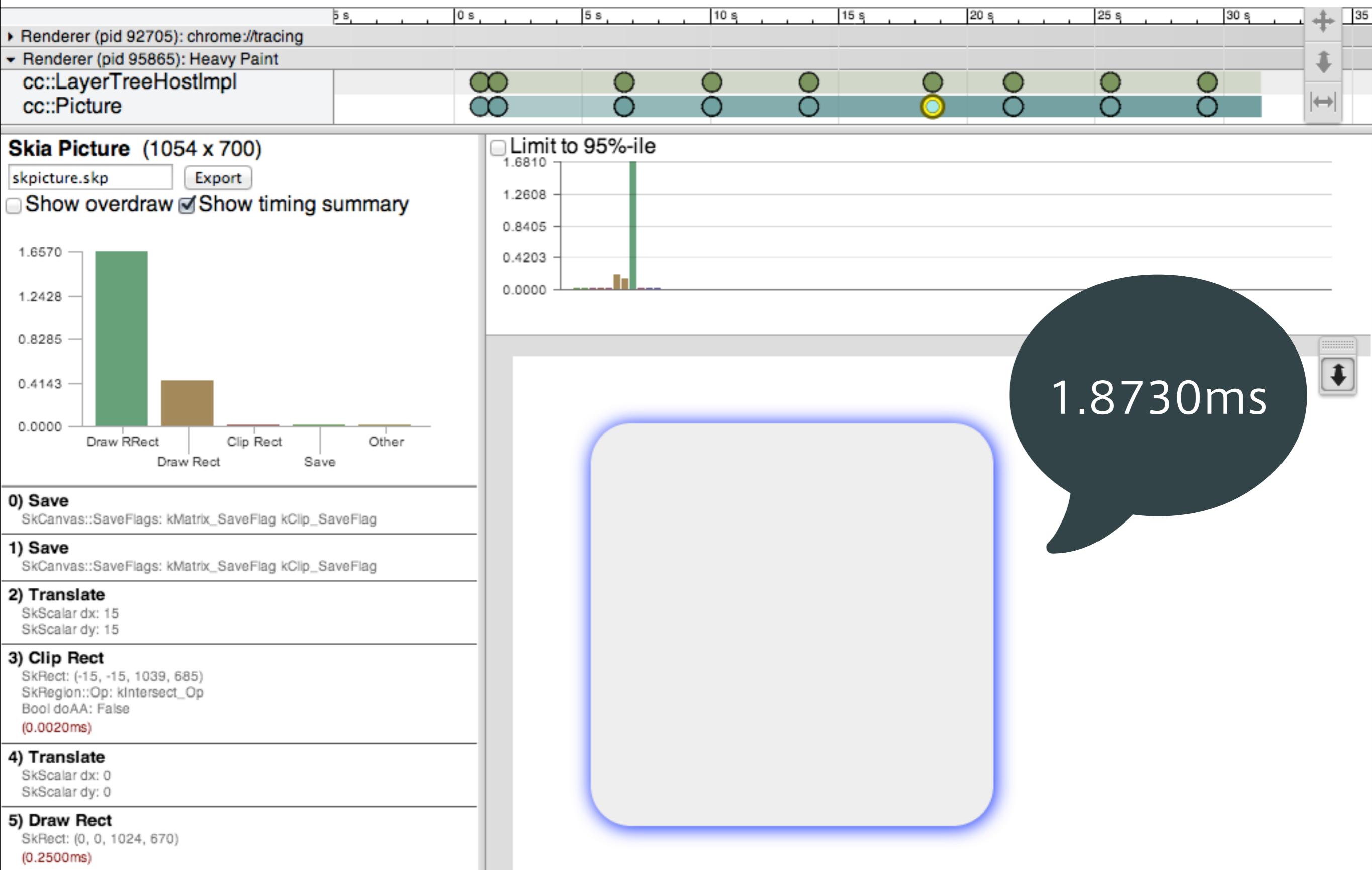
Skia

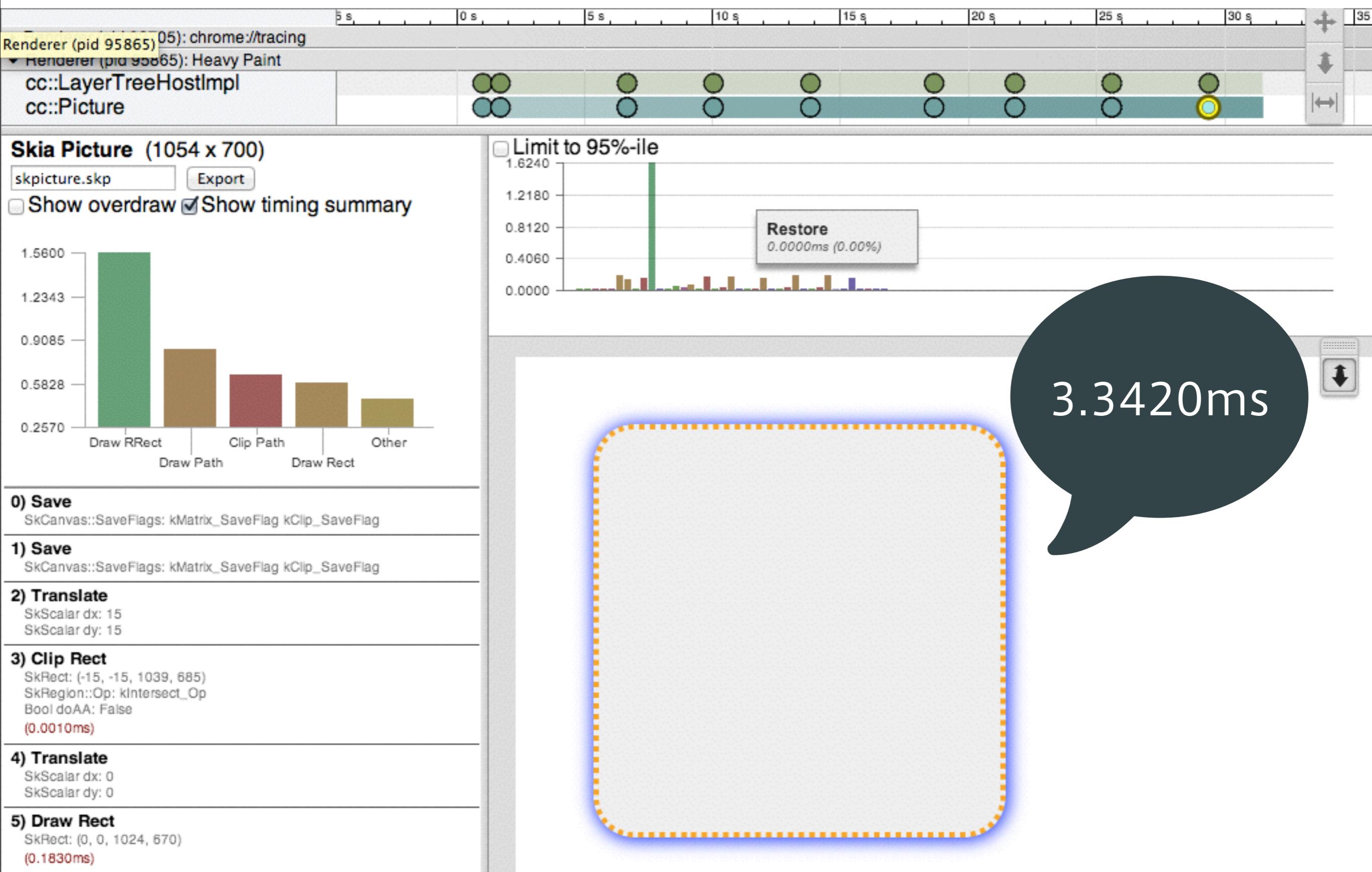
Chromium や Android の 2D描画エンジン

実行された描画コマンドの履歴を見れば

スタイルの複雑さが分かる







Topcoat

CSS for clean and fast web apps.

Demos

Benchmarks

Blog

Download

Dashboard ▼

?

Button on Nexus4

Button

baseline

Button base

Topcoat

Text Input

<http://bench.topcoat.io/>

Search input

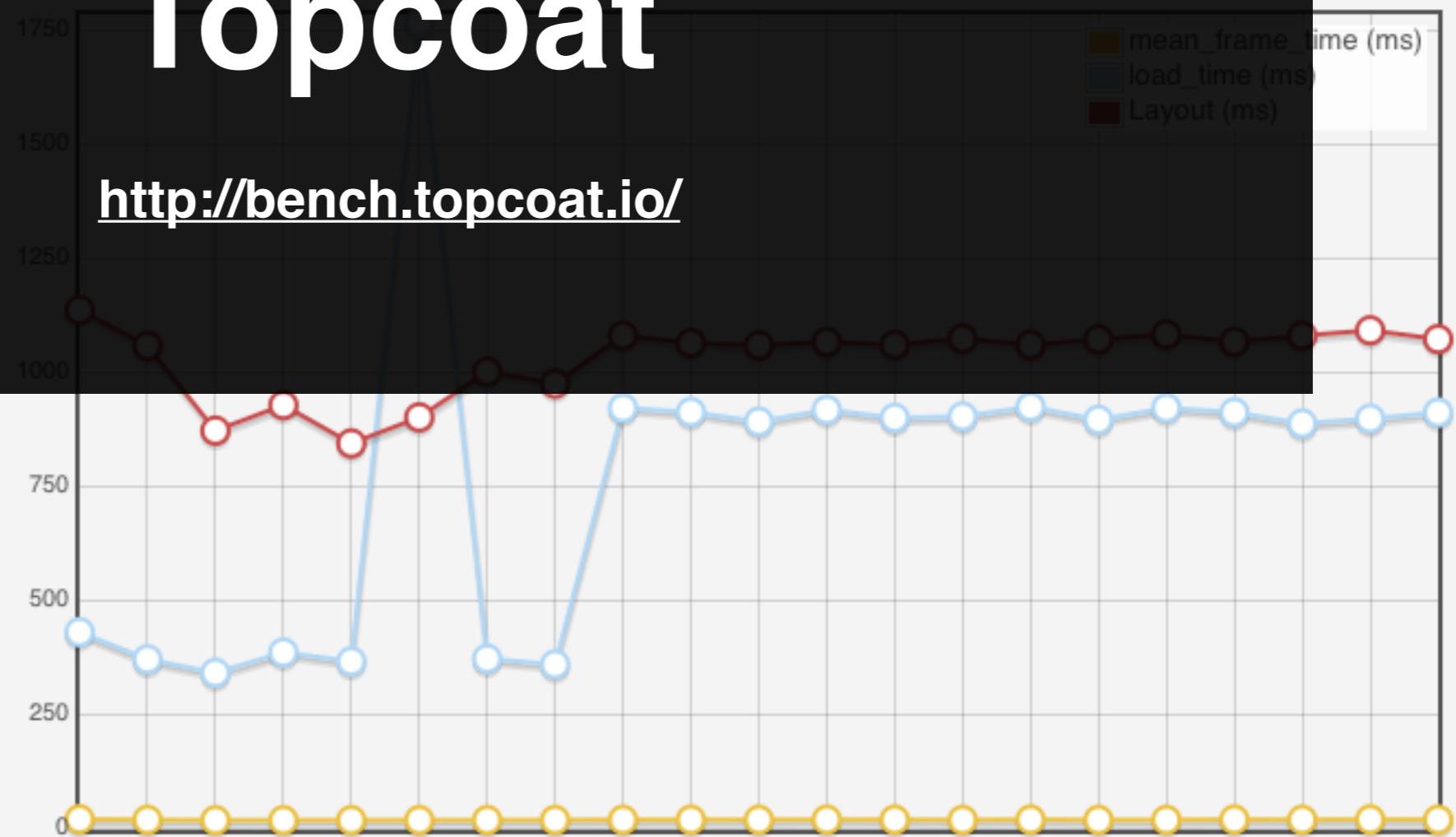
Radio input

Checkbox

Checkbox base

Range input

Input Base



mean_frame_time (ms)

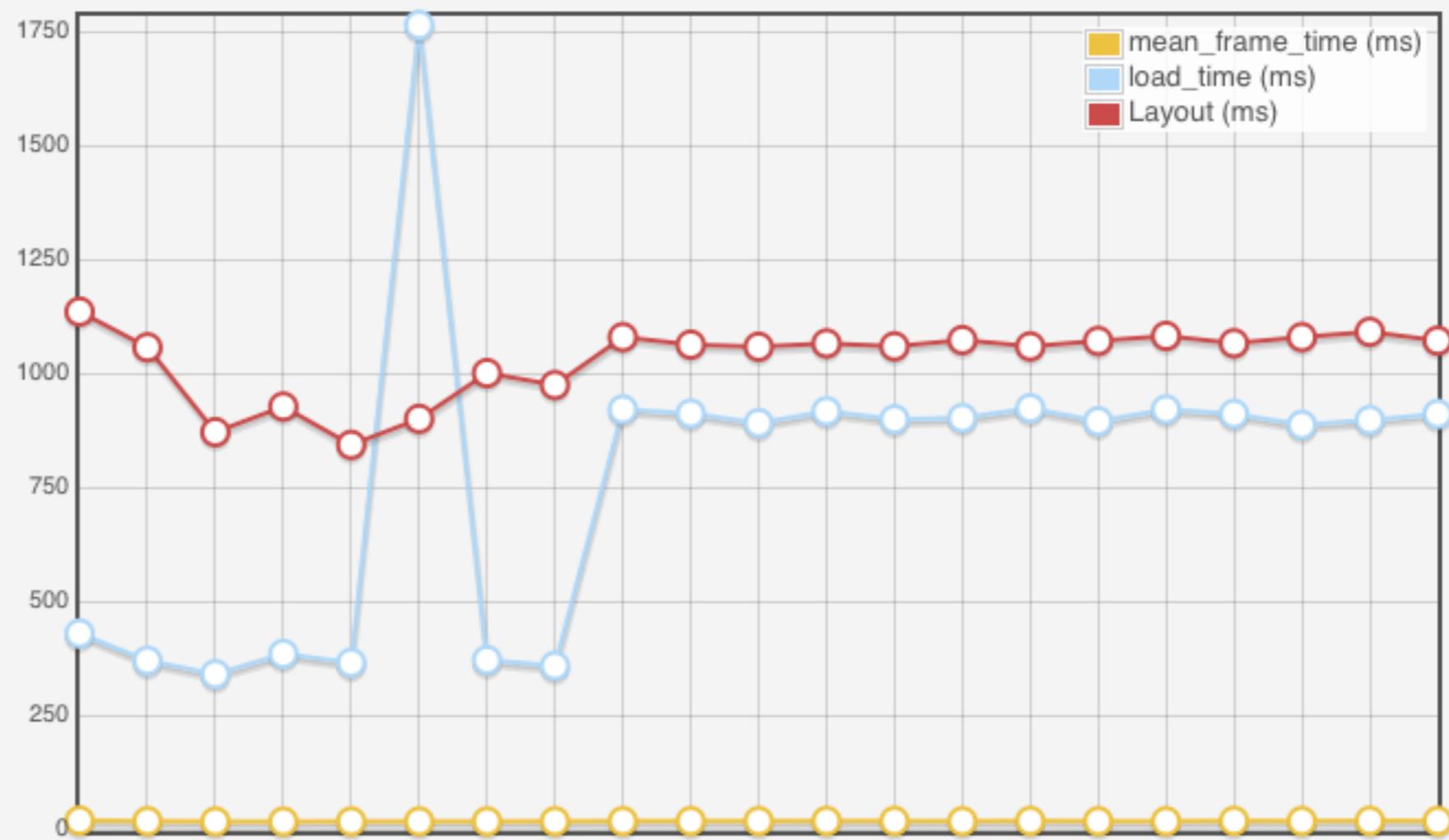
load_time (ms)

Layout (ms)

[Dashboard ▾](#)

?

Button on Nexus4

[Button](#)[baseline](#)[Button base](#)[Text Input](#)[Search input](#)[Radio input](#)[Checkbox](#)[Checkbox base](#)[Range input](#)[Input Base](#) mean_frame_time (ms) load_time (ms) Layout (ms)

The screenshot shows a web browser's developer tools open to the Elements tab. On the left, there is a visual representation of a page consisting of a grid of blue rectangular blocks. Overlaid on this are several green horizontal bars, with the longest one spanning the width of the grid.

In the top right corner of the browser window, there is a status bar displaying "Page paint time (ms) 2.3" and "0.7-3.0".

The Elements tab lists the following elements:

- <!DOCTYPE html>
- ▼ <html lang="ja">
- <head>...</head>
- ▼ <body style>
-

Below the element list, there is a navigation bar with tabs: html, body, and span. The span tab is currently selected. Underneath the tabs, there are buttons for Styles, Properties, DOM, Breakpoints, Event Listener, and a checkbox labeled "Show".

The Styles panel displays the following CSS rules:

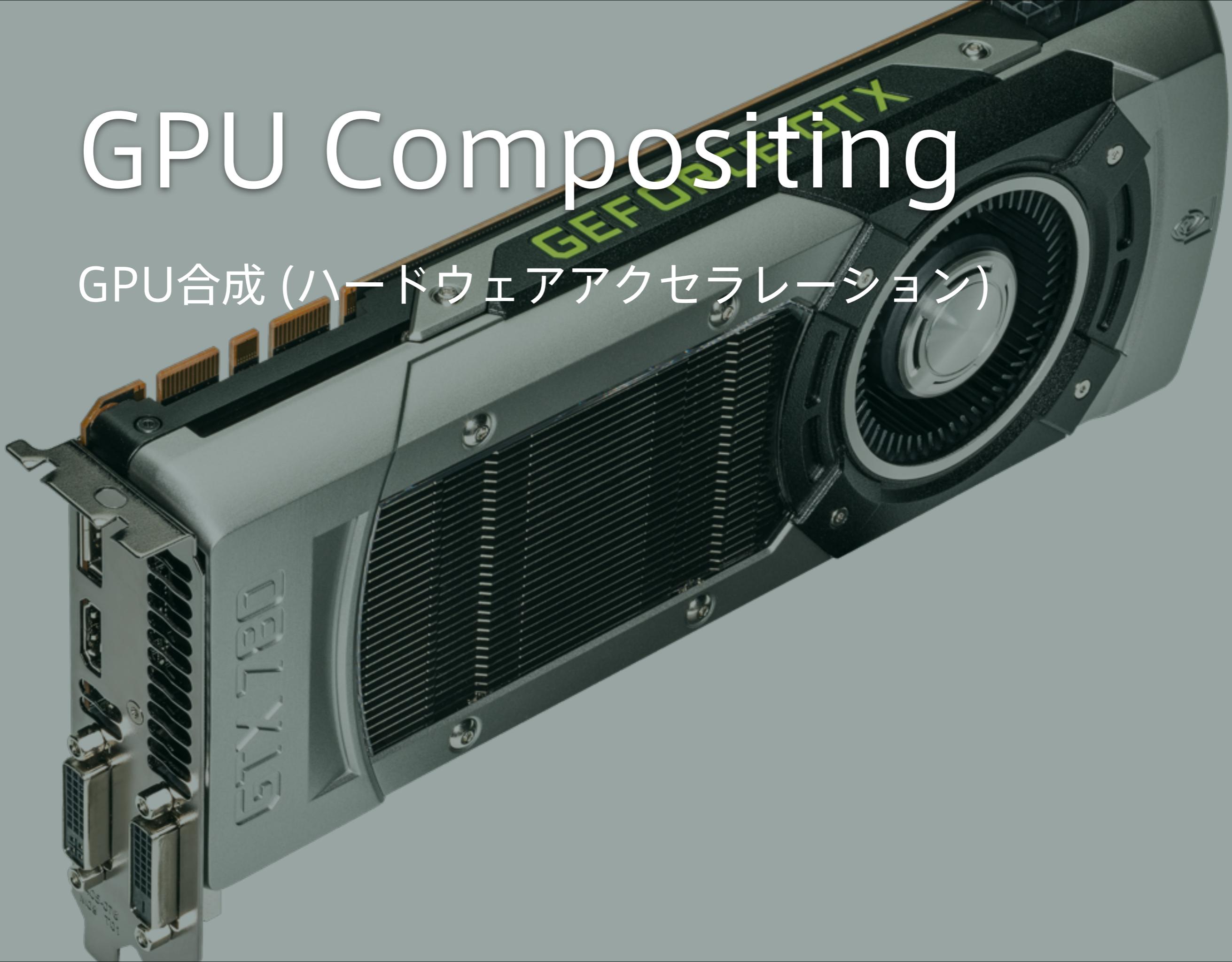
```
element.style { }
span { background: #48a2fb; background: -webkit-linear-gradient(top, #48a2fb, #005cc2); }
```

$+ \alpha$

get more smoothness

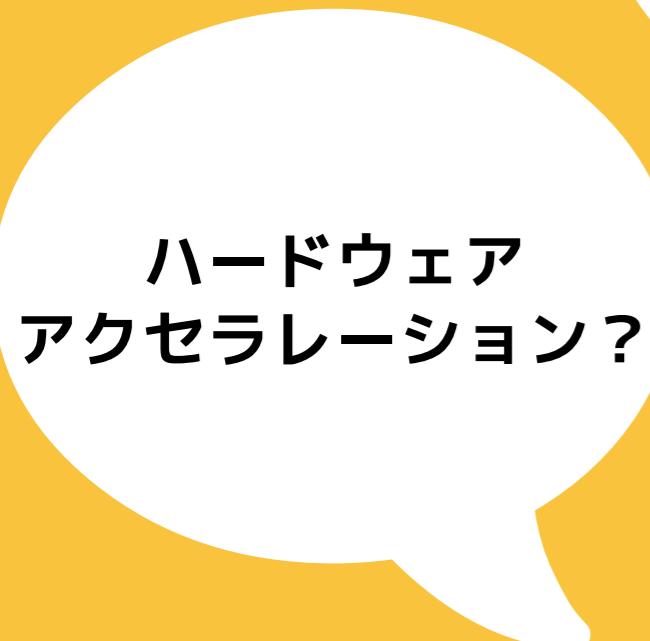
GPU Compositing

GPU合成 (ハードウェアアクセラレーション)

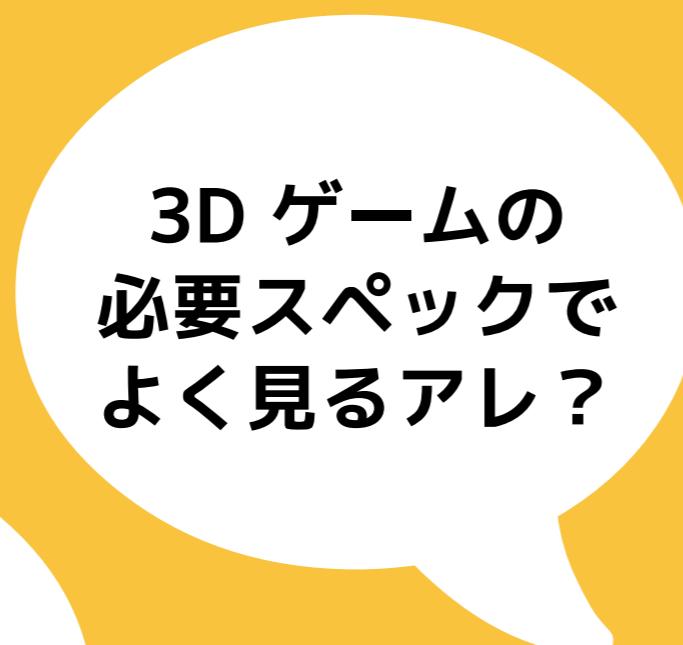


What is GPU ?

画像処理に特化したプロセッサ



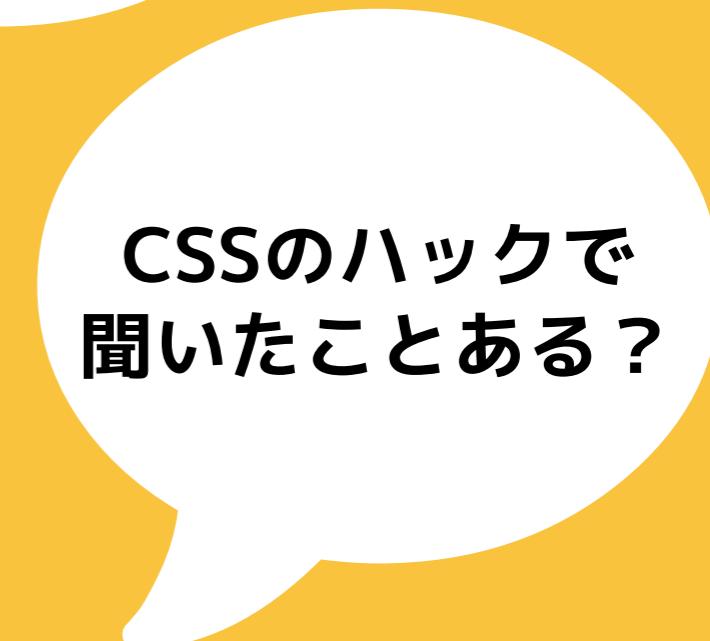
ハードウェア
アクセラレーション？



3D ゲームの
必要スペックで
よく見るアレ？



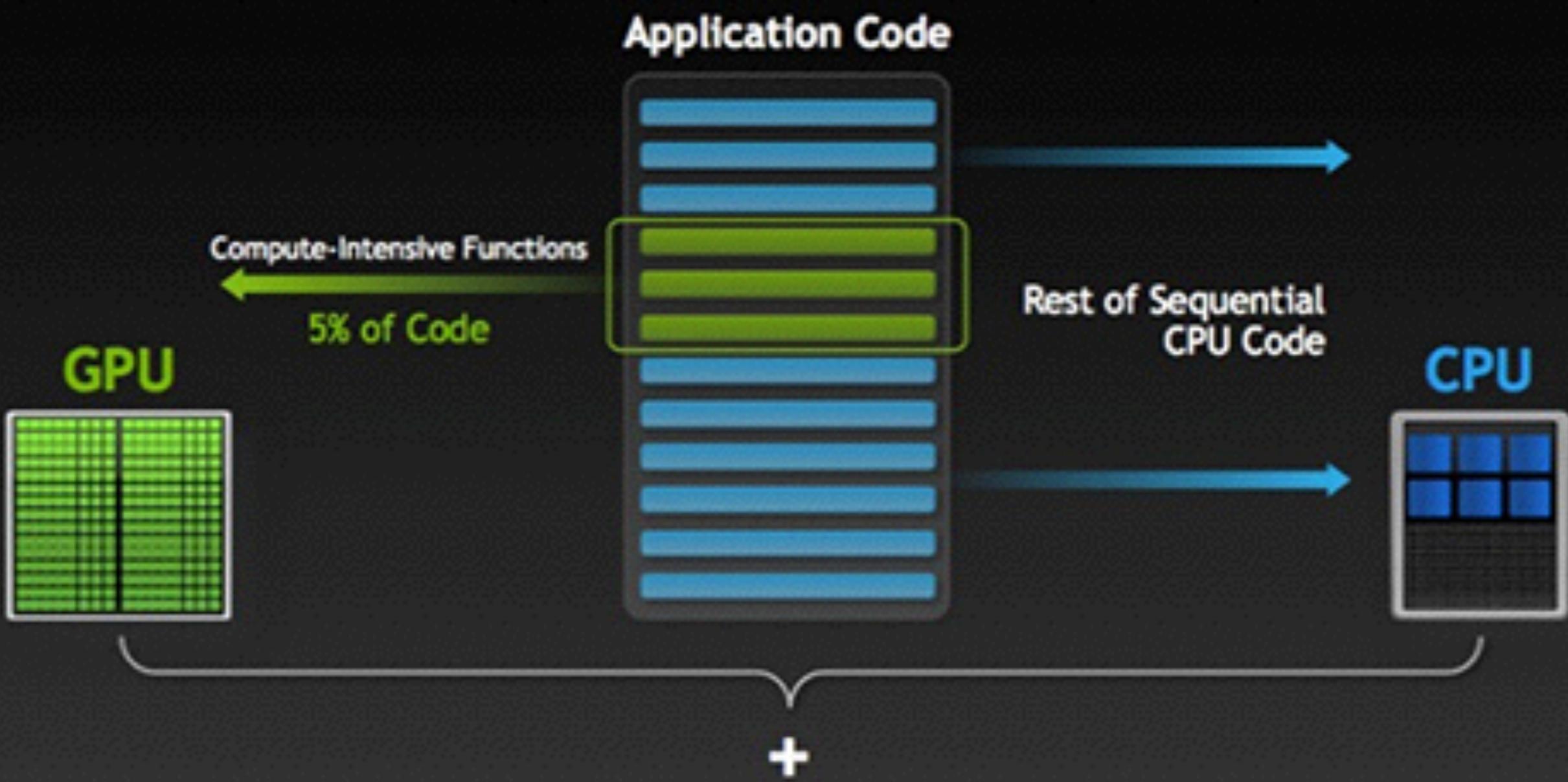
VRAMが
沢山あると
良いヤツ？



CSSのハックで
聞いたことある？



How GPU Acceleration Works





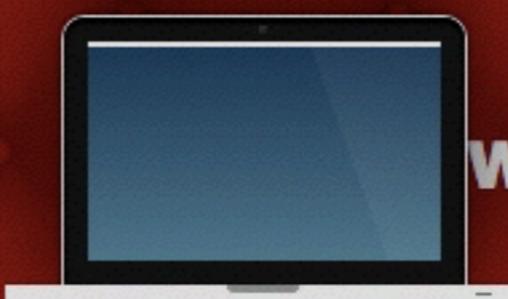
```
28  
29 @keyframes move {  
30   50% {  
31     top: 200px;  
32     left: 200px;  
33   }  
34 }
```

```
35  
36  
37  
38 .macbook {  
39   background-color: #082746;  
40   background-image: linear-gradient(70deg,  
transparent 67%, hsla(0,0%,100%,.05) 67%,
```



add 10 more macbooks

clear

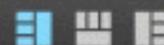


with Top/Left

CodePen PRO is pretty sweet. Just saying.

[Embed This Pen](#)

A Pen by Paul Irish

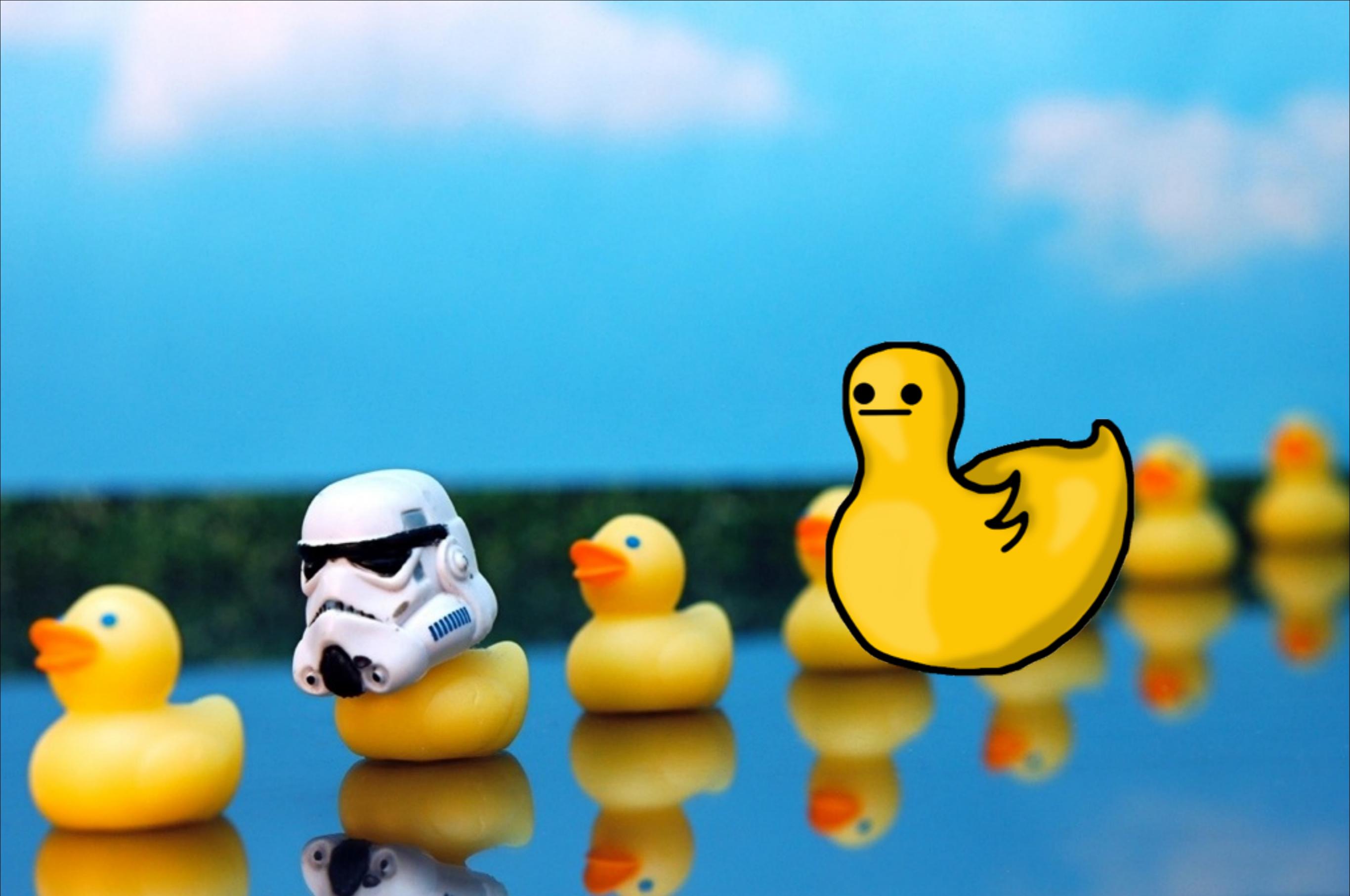
[Shortcuts](#)[Elements](#) [Network](#) [Sources](#) [Timeline](#) [Profiles](#) [Resources](#) [Audits](#) [Console](#) [Layers](#)[!](#) 1 [☰](#) [⚙️](#) [✖️](#)

0 of 0 frames shown

[Events](#)[Frames](#)[Memory](#)

RECORDS

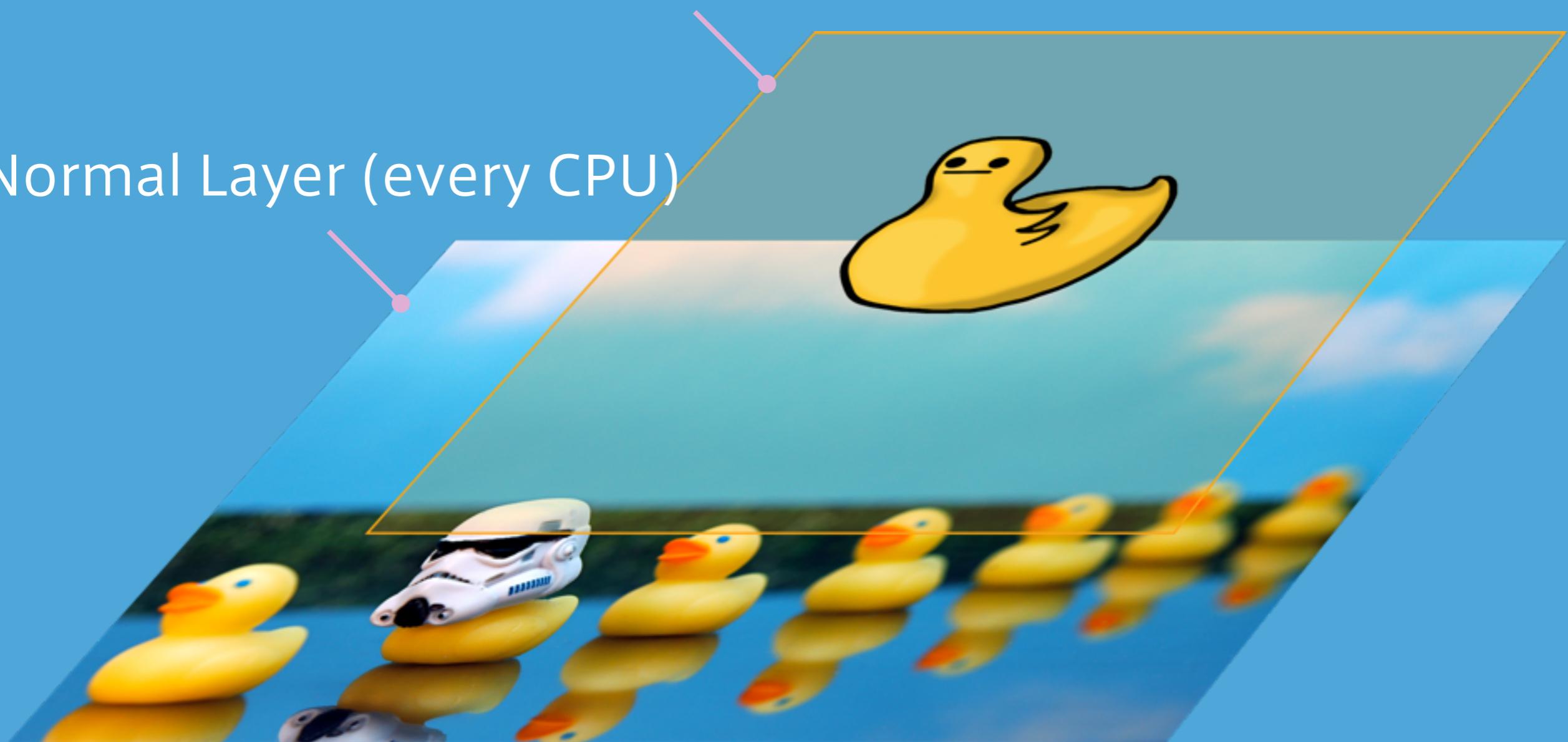
DETAILS



<https://developers.google.com/events/io/sessions/325091862>

Compositing Layer (on GPU)

Normal Layer (every CPU)

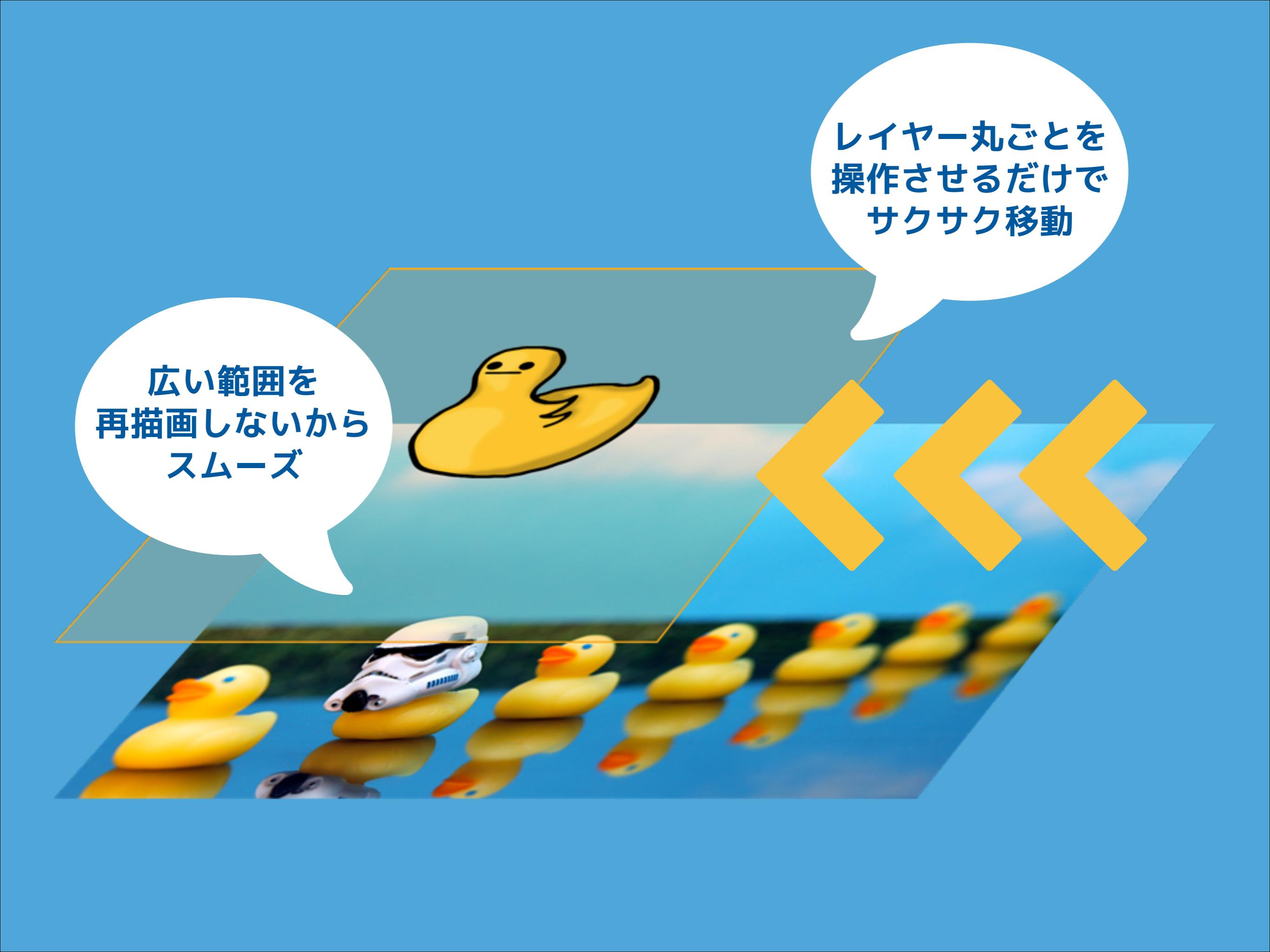


GPU Texture

描画データを
キープしといて
GPU上で合成

更新があれば
描画データを
GPUに再転送





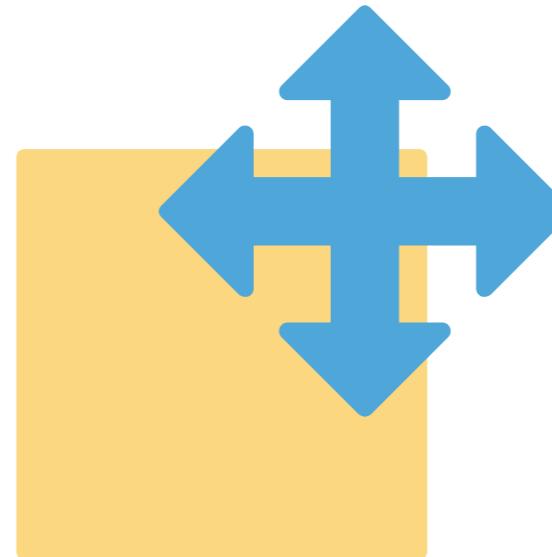
レイヤー丸ごとを
操作させるだけで
サクサク移動

広い範囲を
再描画しないから
スムーズ

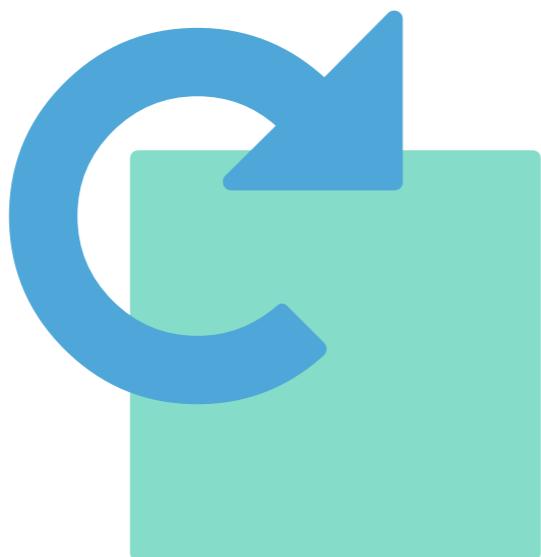
GPUで高速に処理できる (描画データの再転送が不要)



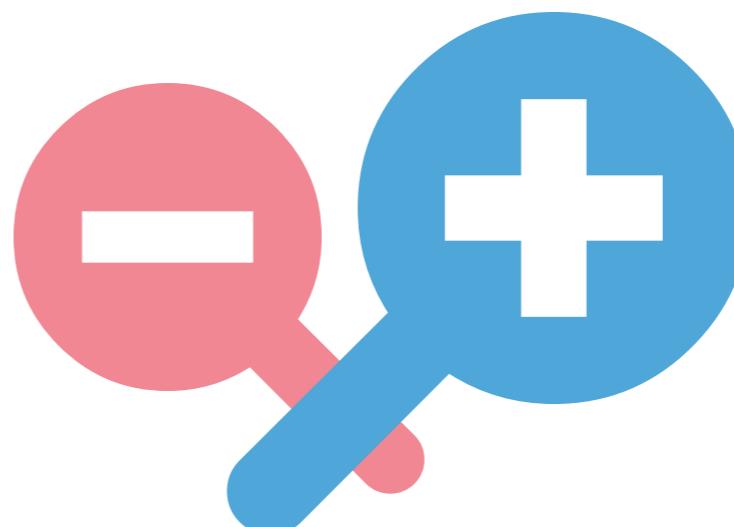
opacity: n



translate(x, y)



rotate(n deg)



scale(n)

Assign-time Layer Promotion

translateZ scaleX rotateX rotateY
rotateZ translate3d scale3d rotate3d
backface-visibility: hidden

Animation Based Layer Promotion

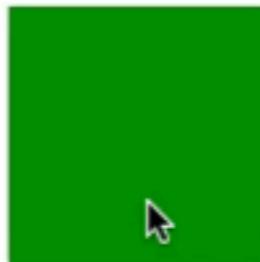
transform: translate transform: opacity

Load-time Layer Promotion

Plugin iFrame Video Canvas WebGL

Show Composited Layer Borders

GPU合成されているレイヤーを可視化する



Accidental layer creation

Click the box above, it spins, woo! Click somewhere else, it stops, boo!

Take a close look at this text when you start and stop the box spinning, it changes ever so slightly, you may have to zoom in to see it. It's losing [subpixel antialiasing](#). This is a symptom of text getting its own texture layer on the GPU.

Fire up this page in [Chrome Canary](#), and turn on "Show composited layer borders" in devtools settings. You'll see the text getting an orange border when the box is spinning, confirming its getting its own texture-backed layer.

Aside from the minor text rendering change, this isn't a big deal on desktop, creating textures and uploading to the GPU is pretty cheap. However, mobile devices often don't have such a friendly relationship with the GPU, and texture creation isn't as cheap.

Why is this happening?

The box has `position: relative`, but so do the headings and paragraphs. This means the headings and paragraphs are layered above the box. When the box gets its own texture-backed layer, by starting a transform-based animation, anything that could appear on top of it must also get a texture-backed layer, as layers without one cannot be rendered on top.

How do I fix it?

Accidental Overlap

レイヤーの意図しない衝突によるCompositing



Accidental layer creation

Click the box above, it spins, woo! Click somewhere else, it stops, boo!

Take a close look at this text when you start and stop the box spinning, it changes ever so slightly, you may have to zoom in to see it. It's losing [subpixel antialiasing](#). This is a symptom of text getting its own texture layer on the GPU.

Fire up this page in [Chrome Canary](#), and turn on "Show composited layer borders" in devtools settings. You'll see the text getting an orange border when the box is spinning, confirming its getting its own texture-backed layer.

Aside from the minor text rendering change, this isn't a big deal on desktop, creating textures and uploading to the GPU is pretty cheap. However, mobile devices often don't have such a friendly relationship with the GPU, and texture creation isn't as cheap.

Why is this happening?

The box has `position: relative`, but so do the headings and paragraphs. This means the headings and paragraphs are layered above the box. When the box gets its own texture-backed layer, by starting a transform-based animation, anything that could appear on top of it must also get a texture-backed layer, as layers without one cannot be rendered on top.



Animation Based Layer

Accidental layer creation

Stacking Context Layer

Click the box above, it spins, woo! Click somewhere else, it stops, boo!

Take a close look at this text when you start and stop the box spinning, it changes ever so slightly, you may have to zoom in to see it. It's losing [subpixel antialiasing](#). This is a symptom of text getting its own texture layer on the GPU.

Fire up this page in [Chrome Canary](#), and turn on "Show composited layer borders" in devtools settings. You'll see the text getting an orange border when the box is spinning, confirming its getting its own texture-backed layer.

Aside from the minor text rendering change, this isn't a big deal on desktop, creating textures and uploading to the GPU is pretty cheap. However, mobile devices often don't have such a friendly relationship with the GPU, and texture creation isn't as cheap.

Why is this happening?

The box has `position: relative`, but so do the headings and paragraphs. This means the headings and paragraphs are layered above the box. When the box gets its own texture-backed layer, by starting a transform-based animation, anything that could appear on top of it must also get a texture-backed layer, as layers without one cannot be rendered on top.

https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Understanding_z_index/The_stacking_context



Accidental layer creation

Click the box above, it spins, woo! Click somewhere else, it stops, boo!

Take a close look at this text when you start and stop the box spinning, it changes ever so slightly, you may have to zoom in to see it. It's losing [subpixel antialiasing](#). This is a symptom of text getting its own texture layer on the GPU.

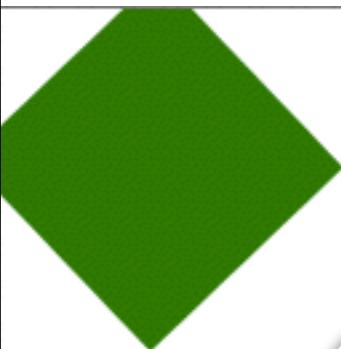
Fire up this page in [Chrome Canary](#), and turn on "Show composited layer borders" in devtools settings. You'll see the text getting an orange border when the box is spinning, confirming its getting its own texture-backed layer.

Aside from the minor text rendering change, this isn't a big deal on desktop, creating textures and uploading to the GPU is pretty cheap. However, mobile devices often don't have such a friendly relationship with the GPU, and texture creation isn't as cheap.

Why is this happening?

The box has `position: relative`, but so do the headings and

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. At the top, there are tabs for 'Elements', 'Network', 'Sources', 'Timeline', 'Profiles', 'Resources', and others. A search bar is located at the top left. Below the tabs, the main pane displays the DOM structure under the '#document' node, which has a size of 469 × 1103. To the right of the DOM tree, a large blue rectangular area represents the visual representation of the document. The bottom right corner of this blue area shows the dimensions 469 × 1103. The overall background of the browser window is white.



Accidental rotation

アニメーション中

Click the box above, or elsewhere, it
stops, boo!

Take a close look at this text when you start and stop the box spinning, it changes ever so slightly, you may have to zoom in to see it. It's losing [subpixel antialiasing](#). This is a symptom of text getting its own texture layer on the GPU.

Fire up this page in [Chrome Canary](#), and turn on "Show composited layer borders" in devtools settings. You'll see the text getting an orange border when the box is spinning, confirming its getting its own texture-backed layer.

Aside from the minor text rendering change, this isn't a big deal on desktop, creating textures and uploading to the GPU is pretty cheap. However, mobile devices often don't have such a friendly relationship with the GPU, and texture creation isn't as cheap.

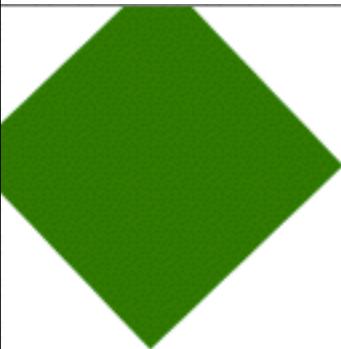
Why is this happening?

The box has `position: relative`, but so do the headings and

The screenshot shows the Chrome DevTools Elements tab with the following details:

- Elements:** A list of elements under the document:
 - #document (469 × 1103)
 - html > body > div.contai...
 - a#edit-with-js-bin (0 × 0)
- Position in parent:** 0,0
- Size:** 469 × 1103

On the right side of the screenshot, there is a diagram illustrating the stacking context. It shows a vertical stack of rectangular layers. From bottom to top, the colors of the layers are blue, grey, blue, grey, blue, grey, blue, grey, blue, grey, blue. The top-most blue layer represents the text element with the orange border, which is part of the '#edit-with-js-bin' element. The other layers represent the background and other elements in the document.



Accidental layer creation

Click the box above, it spins, woo! Click somewhere else, it stops, boo!

Take a close look at this text when you start and stop the box spinning, it changes ever so slightly, you may have to zoom in to see it. It's losing [subpixel antialiasing](#). This is a symptom of text getting its own texture layer on the GPU.

Fire up this page in [Chrome Canary](#), and turn on "Show composited layer borders" in devtools settings. You'll see the text getting an orange border when the box is spinning, confirming its getting its own texture-backed layer.

Aside from the minor text rendering change, this isn't a big deal on desktop, creating textures and uploading to the GPU is pretty cheap. However, mobile devices often don't have such a friendly relationship with the GPU, and texture creation isn't as cheap.

Why is this happening?

The box has `position: relative`, but so do the headings and

A screenshot of the Chrome DevTools Elements tab. At the top, there's a navigation bar with tabs like 'Elements', 'Network', 'Sources', 'Timeline', 'Profiles', 'Resources', and others. Below the navigation bar, the element tree shows the document structure: '#document (469 x 1103) > html > body > div.container > h1'. A large white circle highlights the text 'assumedOverlap' and the Japanese text '重なってるっぽい' (It looks like they overlap). To the right of the circle, there's a 3D-style diagram illustrating the stacking context. It shows several rectangular layers floating in space. Some layers are blue, some are grey, and some have a slight perspective. The layers overlap, with some being positioned higher than others. At the bottom right of the screenshot, there's a summary table with the following data:

Position in parent:	0,0
Size:	469 x 1103



```
<!DOCTYPE html>
<html lang="ja">
  <head>...</head>
  <body id="photos" class="latest" data-access-token="2965044905433113ccda3cd69db9cc779effd7a978252622c65b93d12628cf21" data-client-id="32fd47a7a48abc1aa8272285684e2873474d2d629d0e69e606cc0ab01ec50a1e" data-csrf-token="50A937B1CB788B36F22F1F163F5134CF" data-login="true" data-my-id="260025" style="background-color: #f0f0f0;">
    <header id="amb_header" class="amb_default amb">...</header>
    <div id="pecolly_header">...</div>
    <nav>...</nav>
    <div class="service_info_wrap">...</div>
    <div id="yellow_title_wrap">...</div>
    <section>...</section>
    <section class="js-photolist module-photolist" data-type="latest" data-limit="11" data-first-like="true">
      <div class="js-line_wrap line_wrap">
        <div class="block_cell">
          <div class="bg_cell"></div>
          <a class="item_hit_cell js-flick-link-active" href="http://pecolly.jp/user/photos_detail/1301964" data-photolist-key="first_json" data-detail-id="1301964" data-type="latest" data-type-id data-offset-id="1301964" data-offset="0" data-keyword-sort="0">...</a>
          <div class="photo_cell_wrap" data-photo-id="1301964">...</div>
        <div class="detail_wrap">
          <div class="wrap01">
            <p class="large_pecolly_count js-pecolly">
              <i class="peco_anim_list js-pecolly-animation"></i>
              <i class="pecolly_off js-pecolly-active" data-photo-id="1301964"></i>
              <span class="count js-pecolly-count">...</span>
              <span class="apeeal_wrap js-pecolly-tutorial">
                <i class="appeal_peccolly appeal_peccolly_anim"></i>
                <i class="appeal_peccolly_point"></i>
              </span>
            </p>
            <ul>...</ul>
          </div>
          <div class="wrap02">
            <div class="user_status">
              <div class="user_thumb">
                <a href="http://pecolly.jp/user/profile/85987" class="user_thumb_link">
                  
                </a>
              </div>
              <div class="user_info">
                <a class="name" href="http://pecolly.jp/user/profile/85987">マロン</a>
                <span class="photo_title">牛肉牛蒡巻きなお弁当11/25</span>
              </div>
            </div>
          </div>
        </div>
      </div>
    </section>
  </body>

```

Case Study

ケーススタディ

WRM-SG

72

Jank Busting

パフォーマンスの阻害要因を取り除く



Smooth?

たとえばスクロール時の
ペイントパフォーマンスを例に考える



30FPS

24FPS

30FPS前後でも安定していれば
スムーズな操作感を得ることができる



30FPS

12FPS

Jankポイントがあると
操作の引っかかり・違和感につながる

レンダリングパフォーマンスを落とすJank

不要なペイントにつながる

- position: fixed
- overflow: scroll
- :hover 効果
- touchリスナーの処理

重いペイントにつながる

- 複雑な見た目のCSS
- 画像のデコードとリサイズ
- 大きすぎる無駄な空レイヤー

<http://jankfree.org/jank-busters-io-2013/template.html#15>



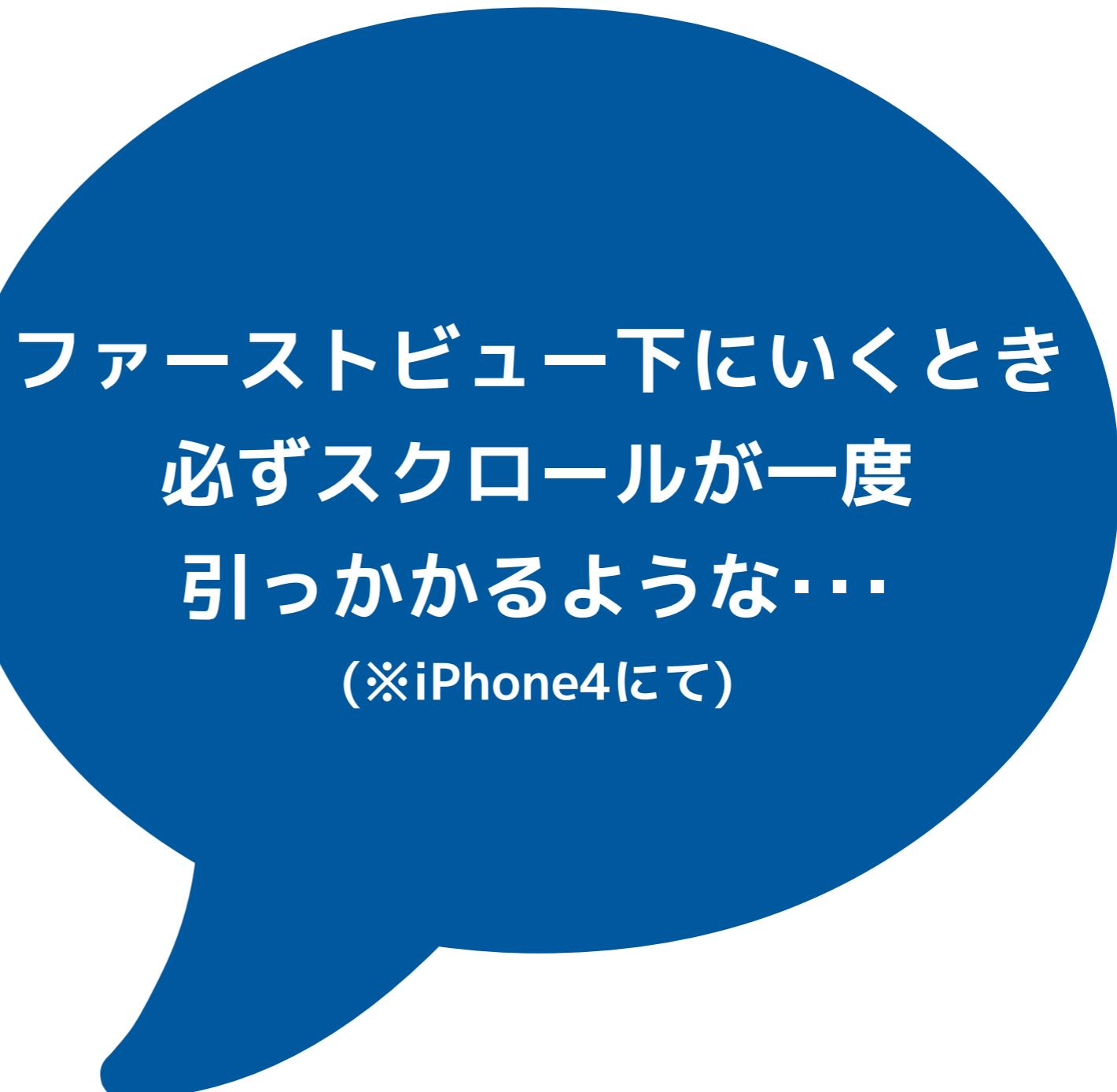
Jank!

Jankポイントを解消するには
原因を特定・対処するノウハウが必要

Jank!

某プロジェクト

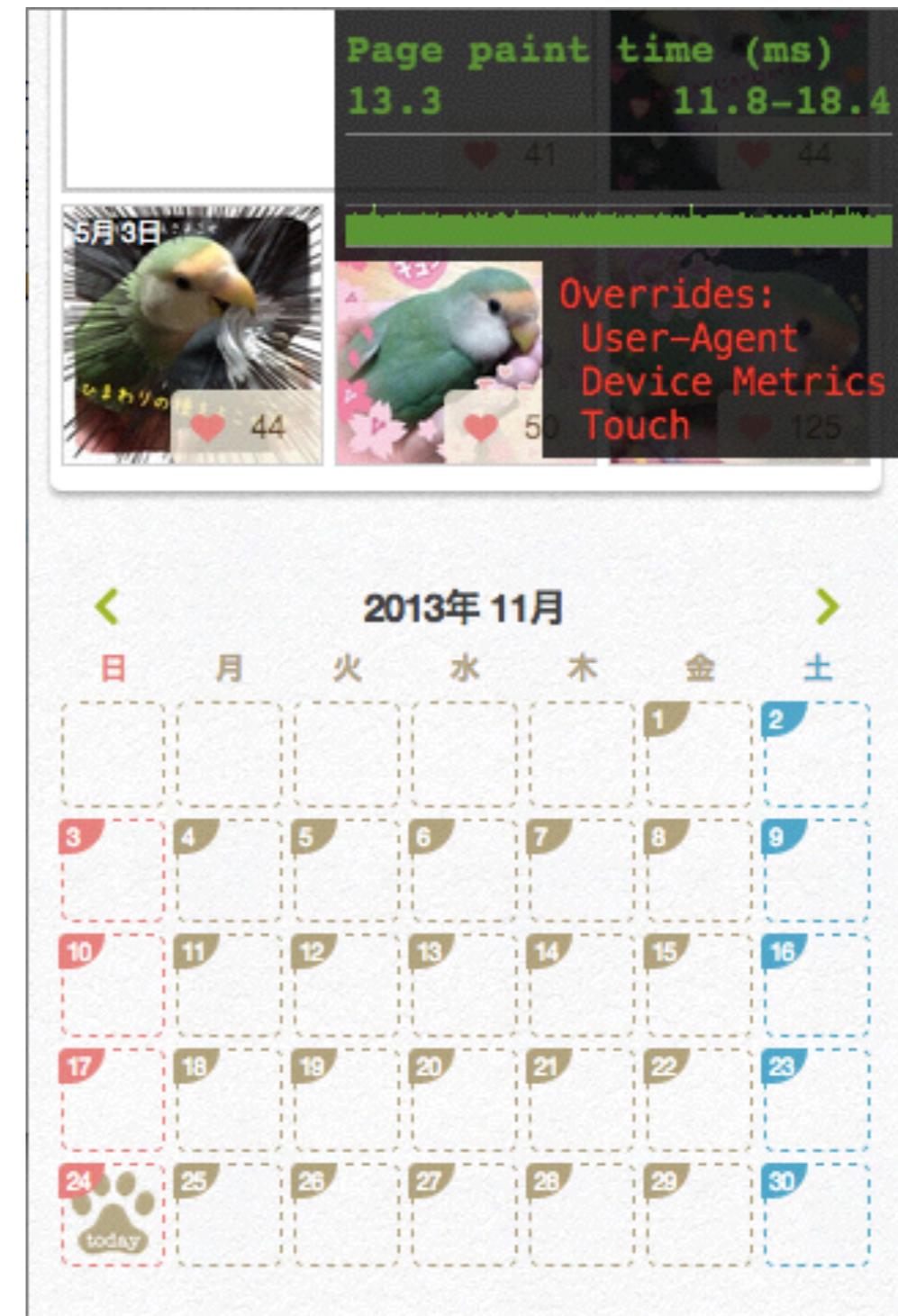




ファーストビュー: 4ms



約1.5画面下: 13.3ms



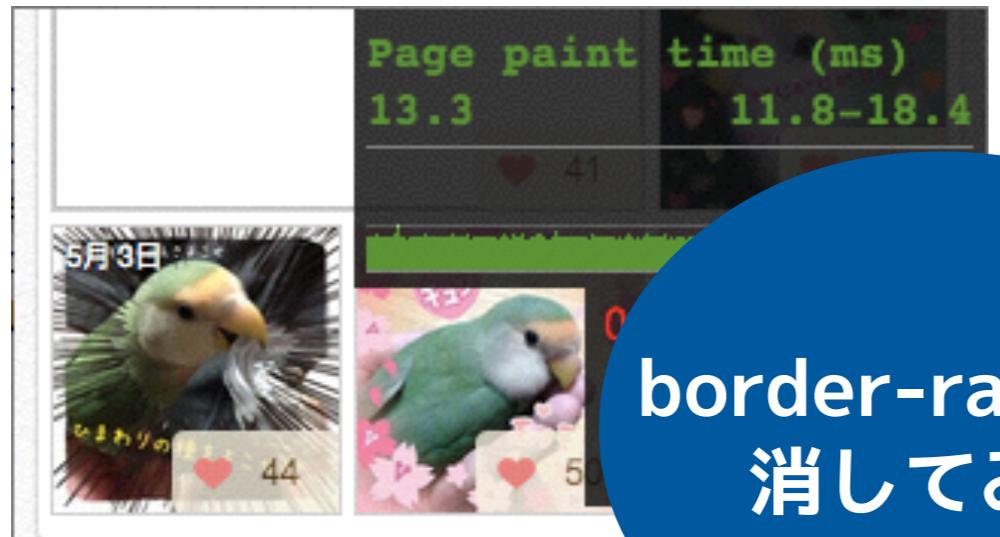
ファーストビュー: 4ms



約1.5画面下: 13.3ms



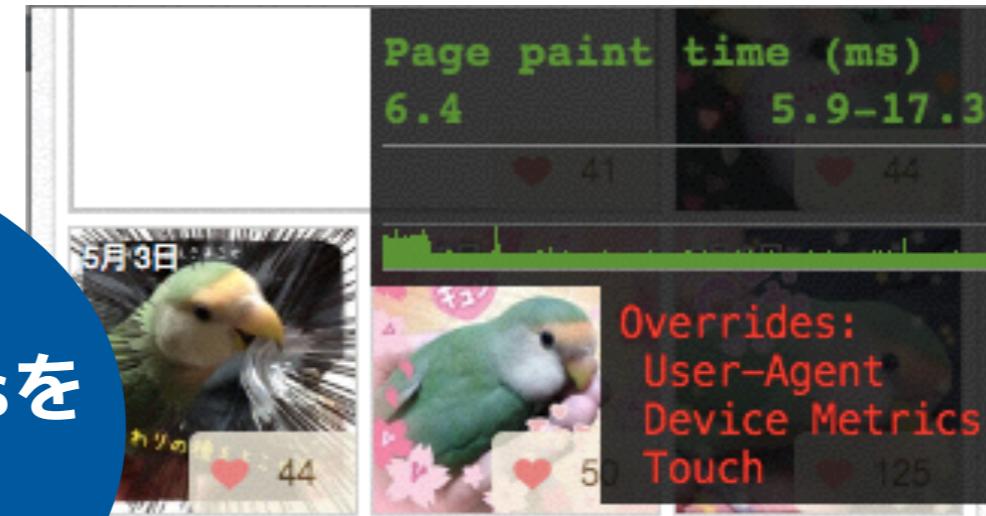
チューニング前: 13.3ms



border-radiusを
消してみた



チューニング後: 6.4ms



CSSの仕事しそう注意

CSSで出来ることから逆算して複雑にするのも危うい

某プロジェクト2



この(グラデや画像が山盛りな)
ページを開いて暫くすると
ブラウザが落ちるんです！

X-)

同僚

後日

余裕なかったので、ややスルー気味対応でした



bodyに おまじないCSSを
適用したら直りました
:-)



同僚

おまじないの内容

```
body {  
  -webkit-backface-visibility:hidden;  
}
```

Oh

$\Sigma(\acute{\Delta};)$

おまじない のろいの内容

```
body {  
  -webkit-backface-visibility:hidden;  
}  
↑ BODYまるごと GPU Compositing してる
```

GPU Texture

更新があれば
描画データを
GPUに再転送

描画データを
キープしといて
GPU上で合成

広域に適用する

常に描画内容が更新される

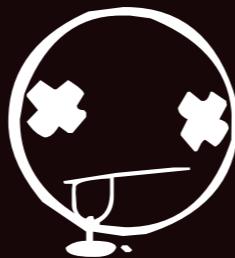
GPUへの再転送コストが高くなる

描画処理に副作用ができる可能性!!

GPUは万能薬ではない

怪しいプロパティで何となく治った = 副作用キケン

某プロジェクト3





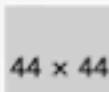
いいねボタン(仮称)につけた
アニメーションの開始に
ラグがあるんだけど…

:-)



Lorem Ipsum is simply dummy text of the printing and typesetting industry.

Like



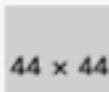
Lorem Ipsum is simply dummy text of the printing and typesetting industry.

Like



Lorem Ipsum is simply dummy text of the printing and typesetting industry.

Like



Lorem Ipsum is simply dummy text of the printing and typesetting industry.

Like

Elements Resources Network Sources Timeline Profiles Audits Console

```
<!DOCTYPE html>
<html slick-uniqueid="3">
  <head>...</head>
  <body>
    <div id="wrapper">
      <div id="head">...</div>
      <div id="tabs" style="height: 404px;">
        <div class="tCont result active" id="result">
          <iframe src="http://fiddle.jshell.net/ahomu/wqXDR/2/show/light/" style="height: 406px;">
        </div>
      </div>
    </div>
  </body>
</html>
```

► Computed Style

Show inherited

▼ Styles

```
element.style {  
}
```

Matched CSS Rules

```
.container div { fiddle.jshell.n...show/light/:31  
  position: relative;  
  text-align: right;  
}
```

```
div {  
  display: block;  
}
```

user agent stylesheet

► Metrics

□ >≡ Q html body #wrapper #tabs #result iframe html body section div



繰り返されたボタンは、すべて同じレイヤーにある



全ボタンが同時に Compositing されていた



Animationのトリガー-classに z-index を加えた



直った(!)



巻き込みによってGPUへのテクスチャ転送量が
多くなったことでラグが発生した？（仮説）

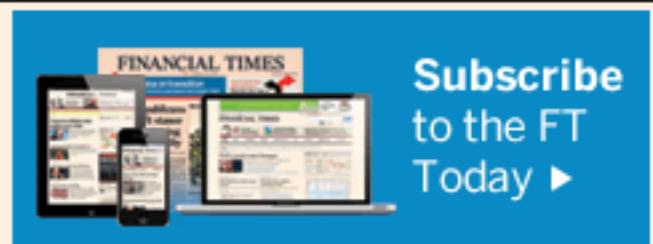
時には対症療法も必要

気づいたときには根治が難しいケースも少なくない

i By continuing to use this site you consent to the use of cookies on your device as described in our [cookie policy](#) unless you have disabled them. You can change your [cookie settings](#) at any time but parts of our site will not function correctly without them.



Lift up
her voice.



ft.com/frontpage UK All times are London time

[Sign in](#) [Site tour](#) [Register](#) [Subscribe](#)

FINANCIAL TIMES

Search articles, quotes and multimedia

Search

Search sponsored by QATAR FINANCIAL CENTRE

[Advanced search](#)

Financial Times

Reach for the tsars
Russia's Romanovs stage a comeback

Davos with jokes
Comedy and economics at Kinnarps' economic fest

Robert Shrimley #BreakingUpOnTwitter
Tim Harford Universal income is not such a silly idea
Chris Cook Only the old can save the doomed youth

Home UK World Companies Tech Business Companies Personal Finance Life & Arts

Video Interactive Blogs News feed Alphaville beyondborders Portfolio Special Reports In depth Today's Newspaper Tools

RSS

Highlights • Style <http://coding.smashingmagazine.com/2013/05/23/building-the-new-financial-times-web-app/> princess'



NEW WORLD

Six powers reach nuclear deal with Iran

UN climate talks end in compromise

Warsaw talks go into overtime due to deadlock over emission cuts

5:53pm

Egypt expels Turkish ambassador

Cairo accuses Turkey of backing anti-government groups

2:02pm

Greek prime minister warns of reform fatigue

FAST FT

MARKETS DATA

Indices Commodities Currencies Bonds

FTSE 100	6,674.30	-0.11%		6,740
S&P 500	1,804.76	+0.50%		6,720
Eurofirst 300	1,296.92	+0.10%		6,700
Nikkei 225	15,381.72	+0.10%		6,680
Shanghai	2,196.38	-0.43%	Mo Tu We Th Fr	6,660

Data delayed by at least 15 minutes

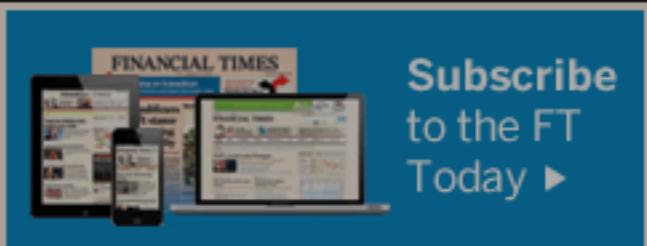
[Portfolio](#) [Markets Data](#)

How do you get from one

By continuing to use this site you consent to the use of cookies on your device as described in our [cookie policy](#) unless you have disabled them. You can change your [cookie settings](#) at any time but parts of our site will not function correctly without them.



Lift up
her voice.



ft.com/frontpage UK All times are London time

[Sign in](#) [Site tour](#) [Register](#) [Subscribe](#)

FINANCIAL TIMES

[Search](#)

[Advanced search](#)



Reach for the tsars
Russia's Romanovs stage a comeback

Davos with jokes

Robert Shimsley #BreakingUpOnTwitter

The last days of the old income is not such a silly idea

England's old can save the doomed youth

Flexible Box Layout Module の利用によって Layout 处理のコストが高くなつた事例

[Home](#) [UK](#) [World](#) [Companies](#) [Markets](#) [Global Economy](#) [Lex](#) [Comment](#) [Management](#) [Personal Finance](#) [Life & Arts](#) [Video](#) [Interactive](#) [Blogs](#) [News feed](#) [Alphaville](#) [beyondbrics](#) [Portfolio](#) [Special Reports](#) [In depth](#) [Today's Newspaper](#) [Tools](#)

Highlights • Style: Baby it's cold outside - Jancis Robinson: Douro wines - Norman Mailer: A Double Life - David Tang - Google's 'security princess'



NEW [WORLD](#)

Six powers reach nuclear deal with Iran

新しい仕様のCSSを導入したが

ブラウザ実装が成熟していなかつたケース

5:53pm

Egypt expels Turkish ambassador

Cairo accuses Turkey of backing anti-government groups

2:02pm

Greek prime minister warns of reform fatigue

over emission cuts

	FTSE 100	6,674.30	-0.11%	6,740
	S&P 500	1,884.76	+0.58%	6,720
	Eurofirst 300	1,296.92	+0.10%	6,700
	Nikkei 225	15,381.72	+0.10%	6,680
	Shanghai	2,196.38	-0.43%	6,660

Mo Tu We Th Fr

Data delayed by at least 15 minutes

[Portfolio](#) [Markets Data](#)

How do you get from one

Flexbox layout isn't slow

By [Paul Irish](#) at 07 October, 2013

g+1 115

performance layout css

TL;DR: Old flexbox (`display: box`) is 2.3x slower than new flexbox (`display: flex`).

A bit ago, Wilson Page wrote a great article for Smashing Magazine digging into how they brought the Financial Times webapp to life. In the article, Wilson notes:

Flexbox layout isn't slow

We spent a good few hours in Chrome Developers Tools' timeline and found the culprit: Shock, horror! — it was our new best friend, flexbox. The timeline showed that some layouts were taking close to 100 milliseconds; reworking our layouts without flexbox reduced this to 10 milliseconds!

<http://updates.html5rocks.com/2013/10/Flexbox-layout-isn-t-slow>

Wilson's comments were about the original (legacy) flexbox that used `display: box;`. Unfortunately they never got a chance to answer if the newer flexbox (`display: flex;`) was faster, but over on CSS Tricks, Chris Coyier opened that question.

We asked Ojan Vafai, who wrote much of the implementation in WebKit & Blink, about the newer flexbox model and implementation.

The new flexbox code has a lot fewer multi-pass layout codepaths. You can still hit multi-pass codepaths pretty easily though (e.g. `flex-align: stretch` is often 2-pass). In general, it should be much faster in the common case, but you can construct a case where it's equally as slow.

That said, if you can get away with it, regular block layout (non-float), will usually be as fast or faster than new flexbox since it's always single-pass. But new flexbox should be faster than using tables or writing custom JS-base layout code.

To see the difference in numbers, I made a head-to-head comparison of old v new syntax.

Old v New Flexbox Benchmark

- [old flexbox vs new flexbox \(with fallback\)](#)

[Store](#)[Mac](#)[iPod](#)[iPhone](#)[iPad](#)[iTunes](#)[Support](#)

iPad mini
with Retina display



iPad Air



iPad Accessories



Compare iPad
Models



iOS 7



iCloud

d Air

of lightness.

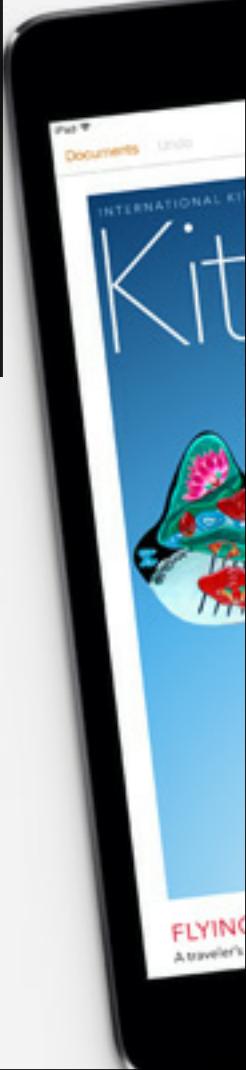
Apple

Page/

iPad mini

with Retina display

Small wonder.



<http://wesleyhales.com/blog/2013/10/26/Jank-Busting-Apples-Home-Page/>



Store

Mac

iPod

iPhone

iPad

iTunes

Support



iPad mini



iPad Air



iPad Accessories



Compare iPad



iOS 7



iCloud

translateZ(0) ハックが、**多数の非アクティブな要素**にも
適用されることで Compositing が高コストになっていた

親要素に必要十分な translateZ(0) ハックがあったので
ネストした要素内の個別ハックを解除して改善した

Small wonder.

Smile. Everyone gets a free terabyte.

<http://code.flickr.net/2013/06/04/adventures-in-jank-busting-parallax-performance-and-the-new-flickr-home-page/>

Biggr. That's right, a terabyte.

Spectaculr. Share in full resolution.

Wherevr. Available anywhere you go.

Forevr. Introducing the all-new Flickr Photo Books.

[Sign In](#)

典型的なスクロールパララックス実装の
FPSをTimeineを元に改善する事例

marginTop や background-position に代わって
translate3d やtranslateZ によるGPU合成で改善

Wherevr. Available anywhere you go.

Forevr. Introducing the all-new Flickr Photo Books.

Sign In



Pinterest でお気に入りの全て（レシピ、記事、旅行のアイデア）を保存しよう！

新規アカウント作成

ログイン

Pinterest

<http://www.smashingmagazine.com/2013/06/10/pinterest-paint-performance-case-study/>

The screenshot shows the Pinterest homepage with a dark background. At the top, there's a navigation bar with a red 'New Account Creation' button and a grey 'Login' button. Below the navigation is a large white 'Pinterest' logo. Underneath the logo is a search bar with the URL 'pinterest.com'. The main content area displays several pins:

- A pin featuring a red windmill with hanging decorations, with the caption "Take a vacation to Taiwan and leave a wish." It has 1,129 repins, 457 likes, and 1 comment.
- A pin showing a room with a fireplace and a dining table, with the caption "Reuse old glass bottles for simple flower vases." It has 1,129 repins, 457 likes, and 1 comment.
- A pin of a person holding a vintage medium format camera, with the caption "Old medium format camera still shoots beautifully." It has 1,129 repins, 457 likes, and 1 comment.
- A pin of a collection of colorful robots, with the caption "Collectible vintage robots." It has 1,129 repins, 457 likes, and 1 comment.

On the left side, there's a 'Recent Activity' sidebar showing pins repinned by users Nadine Harik, Evan Sharp, Ash Huang, and David Gardner. On the right side, there are links for 'Pinterestについて' (About Pinterest), 'ビジネス' (Business), 'デベロッパー' (Developer), '規約 & プライバシー' (Terms & Privacy), and 'サイトマップ' (Site Map). The bottom right corner shows a small image of a person's face.



position: fixed なヘッダーUIに対する
translateZ(0) 適用によるGPU合成の適用

外要素に box-shadow 内要素に border-radius を
分けて適用することで、CSSの描画負荷を軽減した

スクロール時に :hover が反応することによる負荷を
ディレイをかけたり、表現を軽くして対応した

Conclusion

まとめ

おさらい

- ・ まずはタイムラインを見ることから始めよう
- ・ パララックスは⚠に気をつけよう
- ・ CSS3は使いすぎないようにしよう
- ・ GPU Compositing は使い所がキモ！
- ・ 先人の知恵は知っておこう！

まとめ

- ・ パフォーマンスの問題は常にシビア
- ・ 実行環境と共にWeb開発の幅が広がっている
- ・ Web技術のひとつとしてノウハウを深めるべき
- ・ 自分なりにキャッチアップしよう
- ・ よりよいWeb体験を！

Thank you

🏠 <http://aho.mu>

🐦 [@ahomu](#)

🐱 github.com/ahomu



Thanks for amazing photos!

アヒルトルーパー

<http://www.flickr.com/photos/jdhancock/6151250051/>

ぼくらのGeForce

<http://www.flickr.com/photos/gbpublic/8790507077/>

扉画像

<http://www.flickr.com/photos/ryanready/4996206922/>

ワークマン1

<http://www.flickr.com/photos/lac-bac/4678561929/>

ワークマン2

<http://www.flickr.com/photos/lac-bac/4678562003/>

ワークマン3

<http://www.flickr.com/photos/lac-bac/4678561879/>

ワークマン4

<http://www.flickr.com/photos/lac-bac/4679194926/>