REFLOW

Workshop

2019

Vu Phuong Hoang

Main content

- How is a web page rendered ?
- My animation is lag. Why ?
- Similar animation is smooth. How ?

Check this out for some experiments
(github.com/DancingPhoenix88/reflow-workshop)

“Code first, think later !”

- Your fingers -

TOOLS

- Google Chrome (version 73)
- Sublime Text 3 or any text editor with syntax highlighting
- Git or Fork / SourceTree

PART 1

index.html

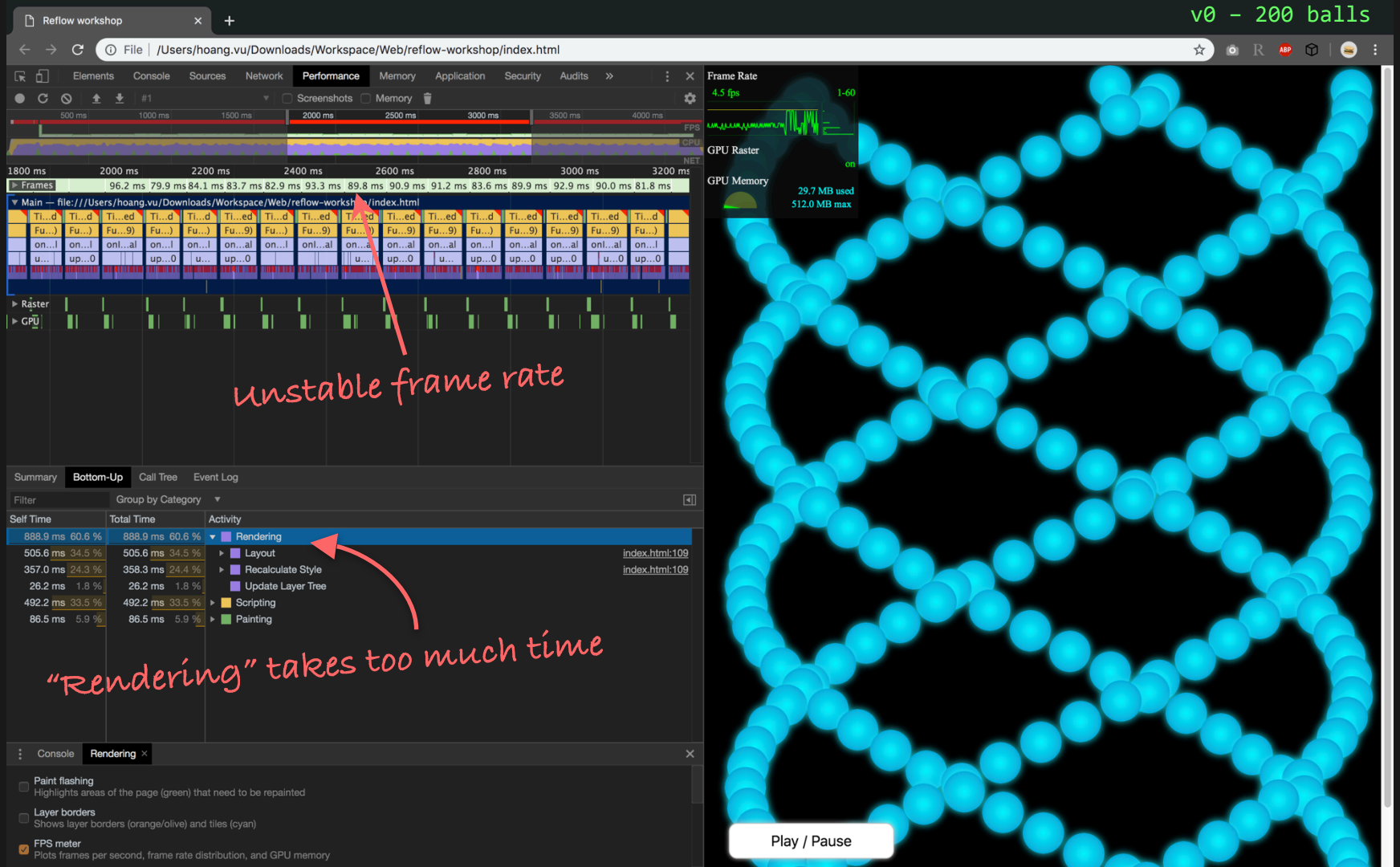
Animate as many balls as possible @ 60 FPS

Version 0

- Branch: feature/animate-position/v0
- Key technique: Update ball positions by ‘`setInterval`’

```
function update_v0 (timestamp) {
  for (var m = 0; m < MOVERS_COUNT; m++) {
    movers[m].style.left = topToLeft(movers[m].offsetTop, timestamp) + 'px';
  }
}
```

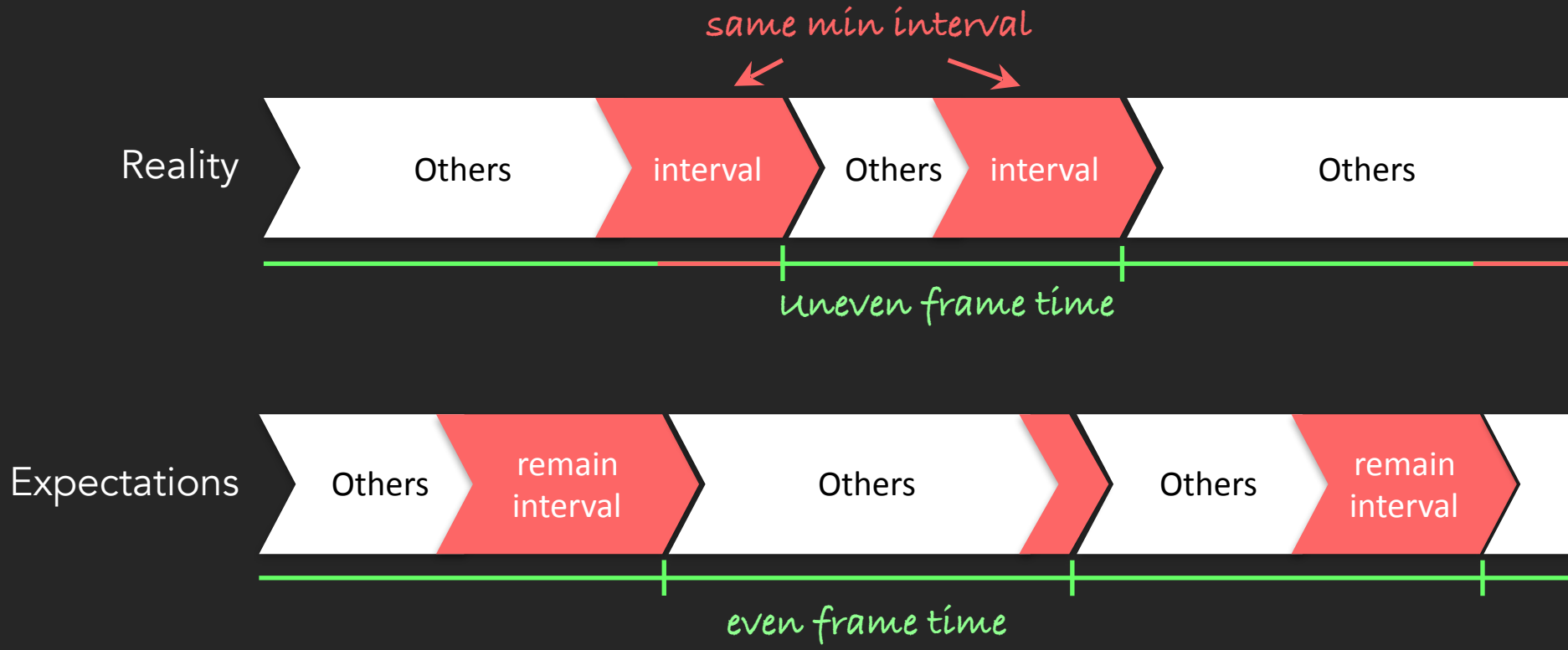
Chrome Devtools


v0 - 200 balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security Audits
Screenshots Memory
Frames
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Raster
GPU
Unstable frame rate
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
888.9 ms 60.6 % 888.9 ms 60.6 % Rendering
505.6 ms 34.5 % 505.6 ms 34.5 % Layout index.html:109
357.0 ms 24.3 % 358.3 ms 24.4 % Recalculate Style index.html:109
26.2 ms 1.8 % 26.2 ms 1.8 % Update Layer Tree
492.2 ms 33.5 % 492.2 ms 33.5 % Scripting
86.5 ms 5.9 % 86.5 ms 5.9 % Painting
"Rendering" takes too much time
Console Rendering
Paint flashing
Highlights areas of the page (green) that need to be repainted
Layer borders
Shows layer borders (orange/olive) and tiles (cyan)
FPS meter
Plots frames per second, frame rate distribution, and GPU memory
Frame Rate
4.5 fps
1-60
GPU Raster
on
GPU Memory
29.7 MB used
512.0 MB max
Play / Pause

Version 0

- Branch: feature/animate-position/v0
- Key technique: Update ball positions by '`setInterval`'
- Problems:
  - Unstable frame rate
  - Rendering activities (Recalculate style, Layout) take too much time
  - Can't reach 60FPS with just 100-200 balls

Unstable frame rate ?



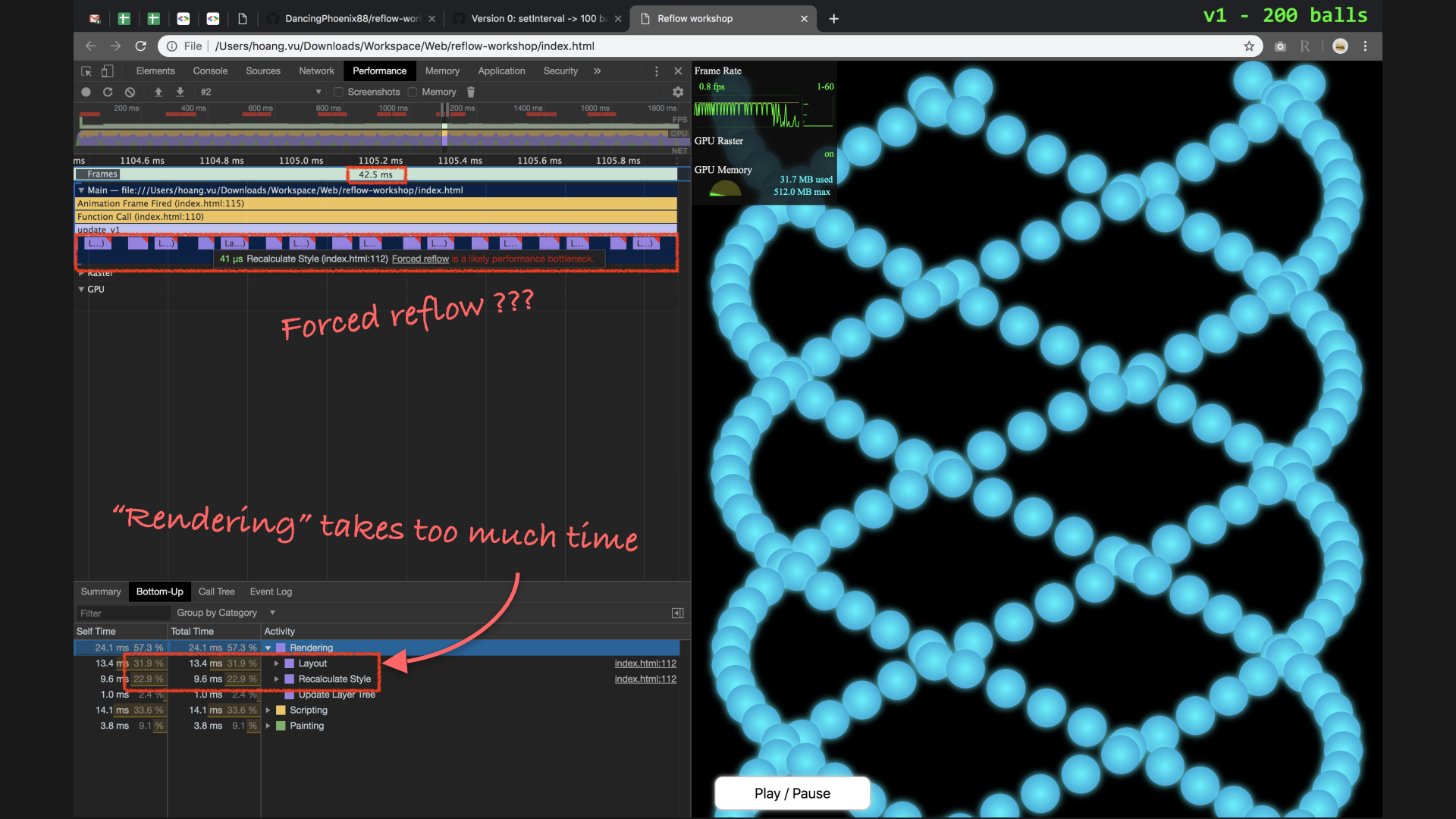
requestAnimationFrame

Version 1

- Branch: feature/animate-position/v1
- Key technique: Update ball positions by '`requestAnimationFrame`'

```
function update_v1 (timestamp) {
  for (var m = 0; m < MOVERS_COUNT; m++) {
    movers[m].style.left = topToLeft(movers[m].offsetTop, timestamp) + 'px';
  }
//if (window.running) setTimeout( onInterval, 1000.0 / 60 );
  if (window.running) rAF( updateFunc );
}
```

Replace 'setInterval' by 'setTimeout' in v.0b


v1 - 200 balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Frame Rate
0.8 fps
1-60
GPU Raster
on
GPU Memory
31.7 MB used
512.0 MB max
Frames 42.5 ms
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Animation Frame Fired (index.html:115)
Function Call (index.html:110)
update_v1
41 µs Recalculate Style (index.html:112) Forced reflow is a likely performance bottleneck.
GPU
Forced reflow ???
"Rendering" takes too much time
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
24.1 ms 57.3 % 24.1 ms 57.3 % Rendering
13.4 ms 31.9 % 13.4 ms 31.9 % Layout index.html:112
9.6 ms 22.9 % 9.6 ms 22.9 % Recalculate Style index.html:112
1.0 ms 2.4 % 1.0 ms 2.4 % Update Layer Tree
14.1 ms 33.6 % 14.1 ms 33.6 % Scripting
3.8 ms 9.1 % 3.8 ms 9.1 % Painting
Play / Pause

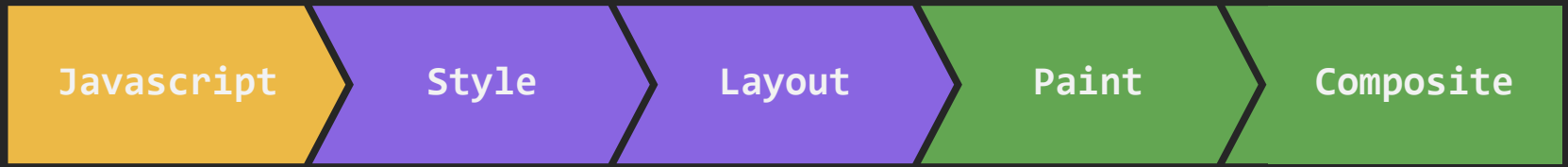
Version 1

- Branch: feature/animate-position/v1
- Key technique: Update ball positions by '`requestAnimationFrame`'
- Problems:
  - ~~Unstable frame rate~~
  - Rendering activities (Recalculate style, Layout) take too much time
  - Can't reach 60FPS with just 100-200 balls

“~~Code first, think later !~~”

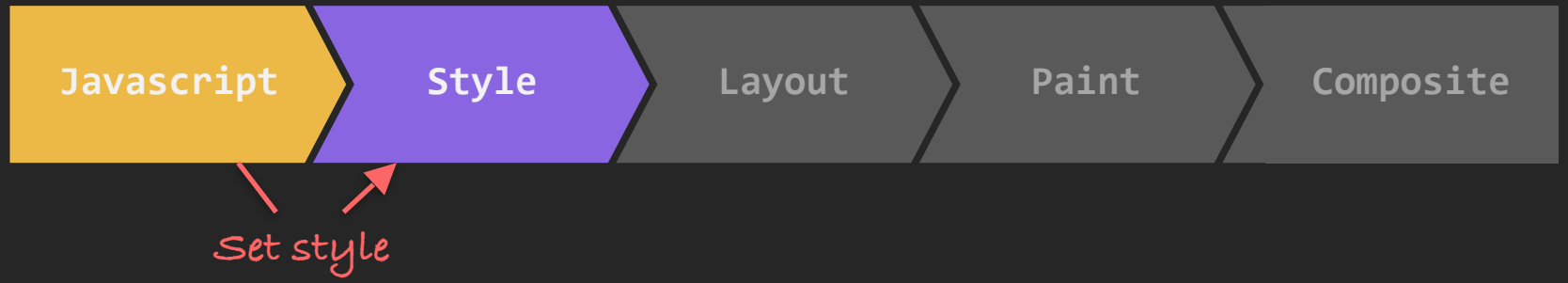
- Your fingers -

Web Rendering Pipeline



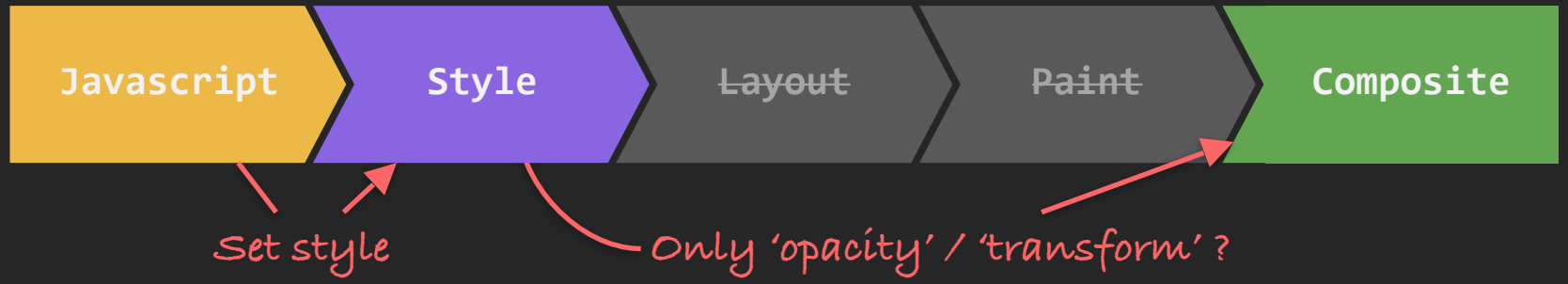
- Javascript: Execute JS code to interact with DOM tree
- Style: Calculate style in CSSOM tree to apply to DOM tree
- Layout: Calculate position & size to arrange elements in the web page
- Paint: Draw elements into layers
- Composite: Apply final transformations to layers and draw to screen

Web Rendering Pipeline



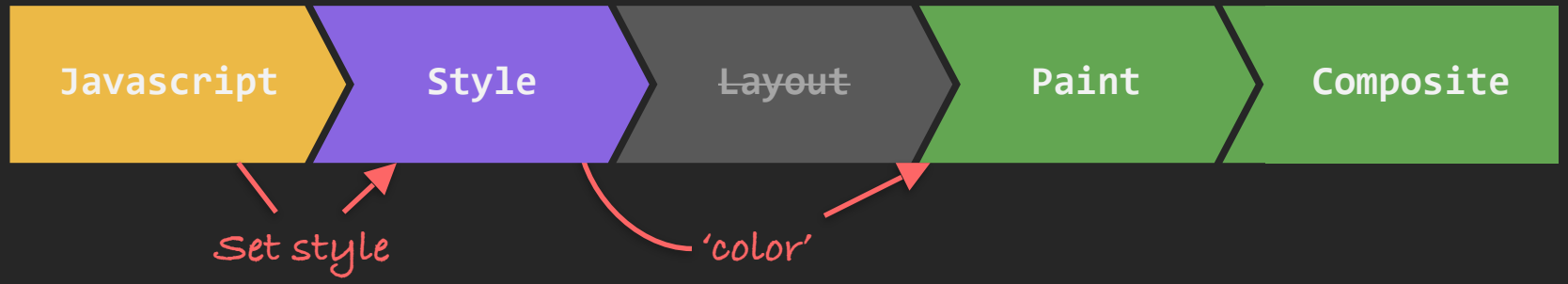
- First frame: Full pipeline to render
- Next frames: Depends on Javascript

Recomposite



If only 'opacity' & 'transform' were changed → re-composite only (BEST)

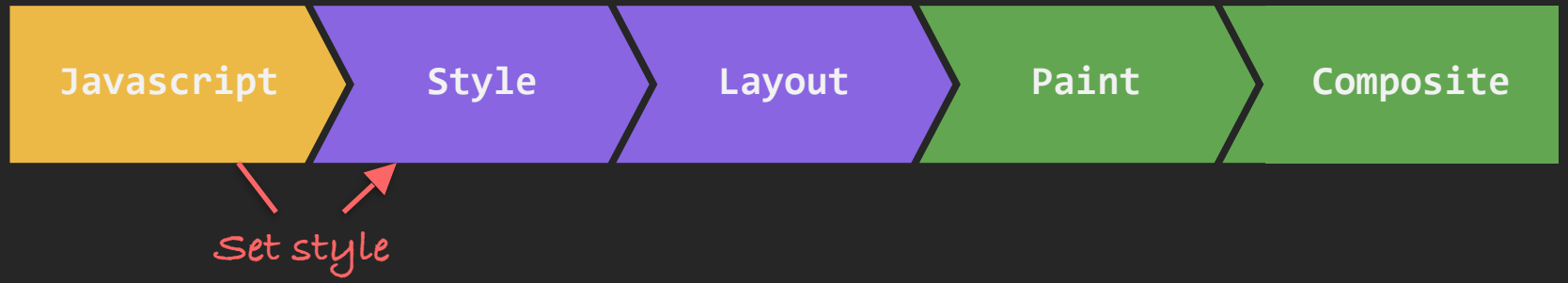
Repaint



If “paint” attributes (color, background-image, shadow …) were changed →re-paint

- Re-paint can be heavy if too many changes
- Re-paint is slow on mobile devices

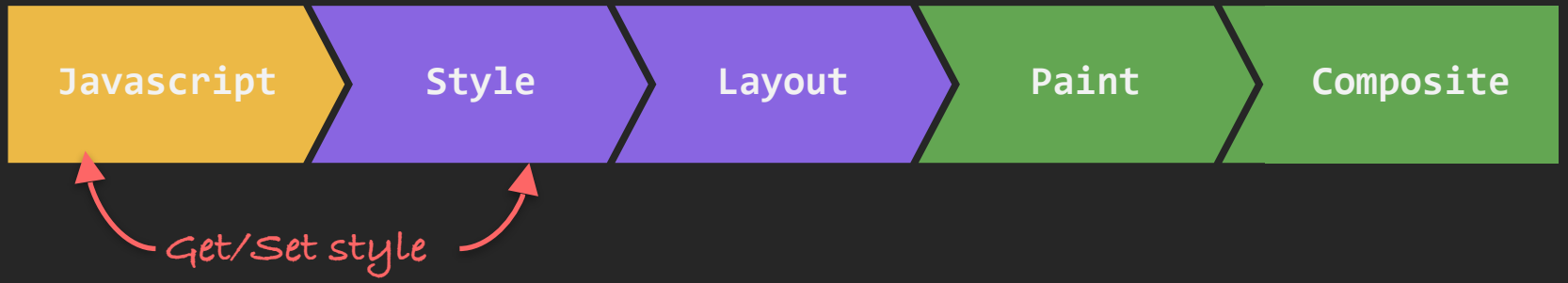
Reflow / Relayout



If “layout” attributes (margin, width, ...) were changed → re-layout (full pipeline)

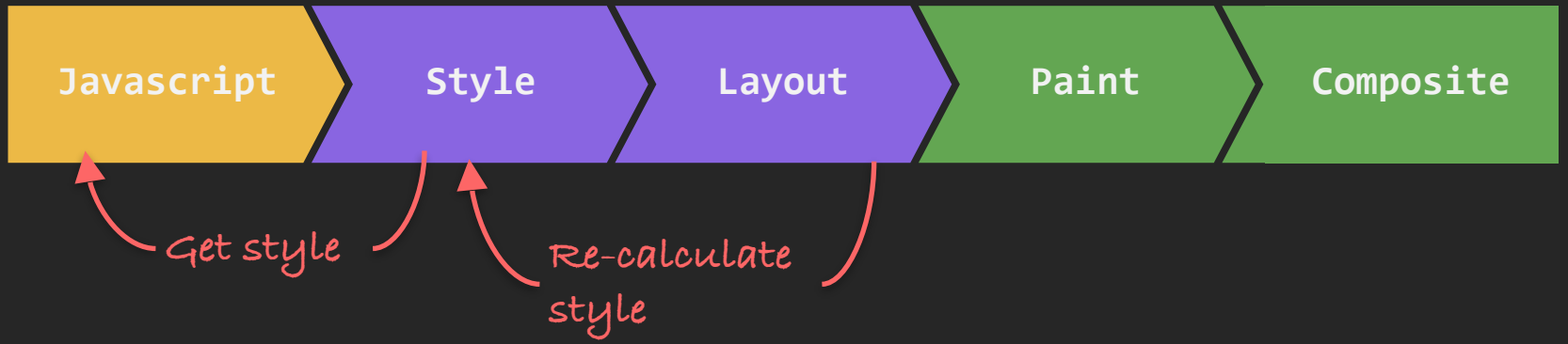
- Reflow leads to repaint & recomposite → Slow
- But browser will wait to perform reflow in batch → Great

Restyle



- When you update style → style needs to be re-calculated
- When you get style → style MIGHT need to be re-calculated due to reflow

Forced reflow / Layout thrashing



- Sometimes you force browser to reflow to get the latest attributes
- This is called “Forced reflow” → Very very bad
- Example: e0 changes layout, e1 get ‘offsetTop’

Version 1

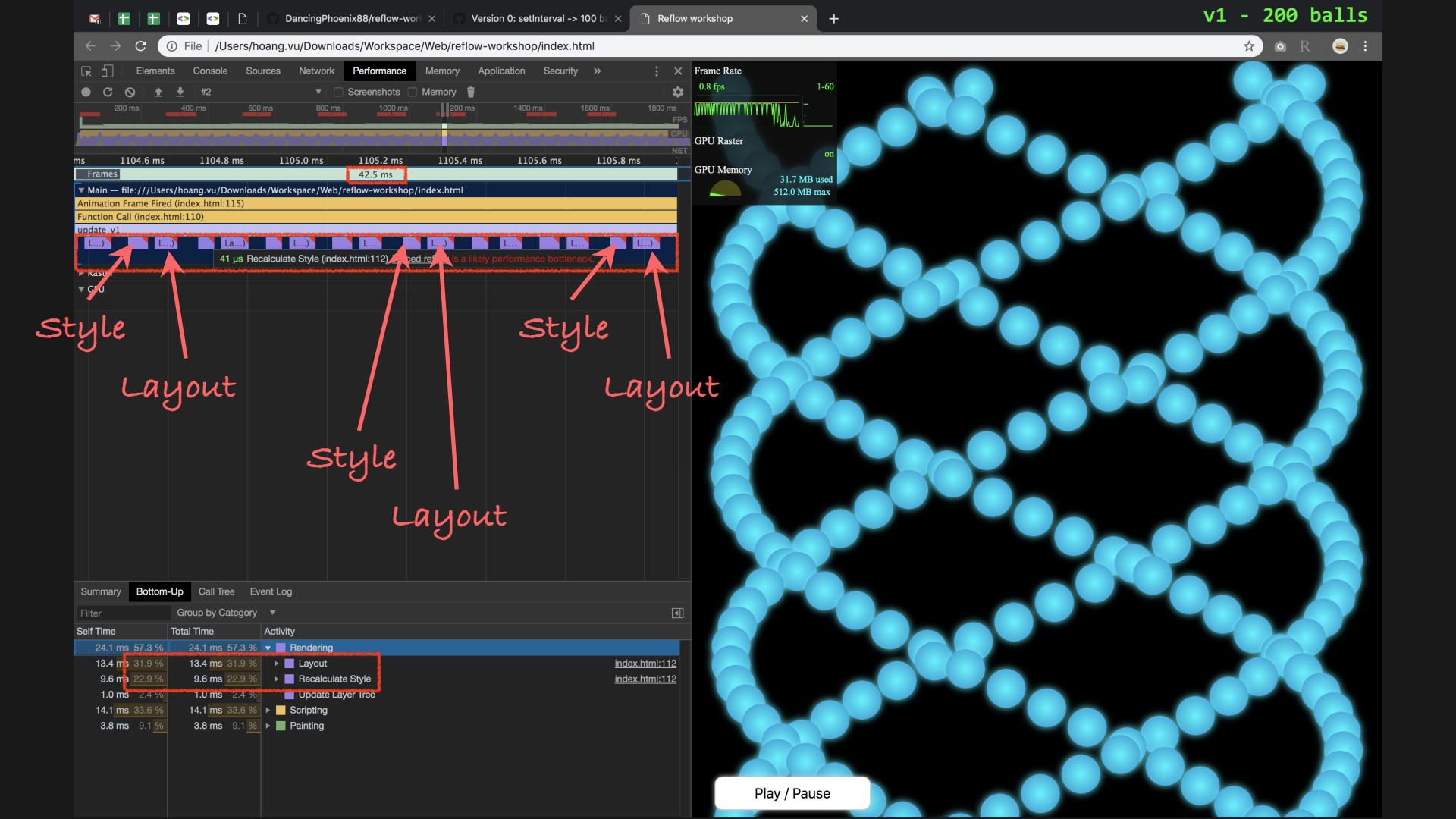
- Branch: feature/animate-position/v1
- Key technique: Update ball positions by '`requestAnimationFrame`'

```
function update_v1 (timestamp) {
  for (var m = 0; m < MOVERS_COUNT; m++) {
    movers[m].style.left = topToLeft(movers[m].offsetTop, timestamp) + 'px';
  }
  if (window.running) rAF( updateFunc );
}
```

Set layout attribute

Get layout attribute

FORCED REFLOWS !!!


v1 - 200 balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
Frames 42.5 ms
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Animation Frame Fired (index.html:115)
Function Call (index.html:110)
update v1
41 µs Recalculate Style (index.html:112)
Style
Layout
Style
Layout
Style
Layout
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
24.1 ms 57.3 % 24.1 ms 57.3 % Rendering
13.4 ms 31.9 % 13.4 ms 31.9 % Layout index.html:112
9.6 ms 22.9 % 9.6 ms 22.9 % Recalculate Style index.html:112
1.0 ms 2.4 % 1.0 ms 2.4 % Update Layer Tree
14.1 ms 33.6 % 14.1 ms 33.6 % Scripting
3.8 ms 9.1 % 3.8 ms 9.1 % Painting
Frame Rate
0.8 fps
1-60
GPU Raster
on
GPU Memory
31.7 MB used
512.0 MB max
Play / Pause

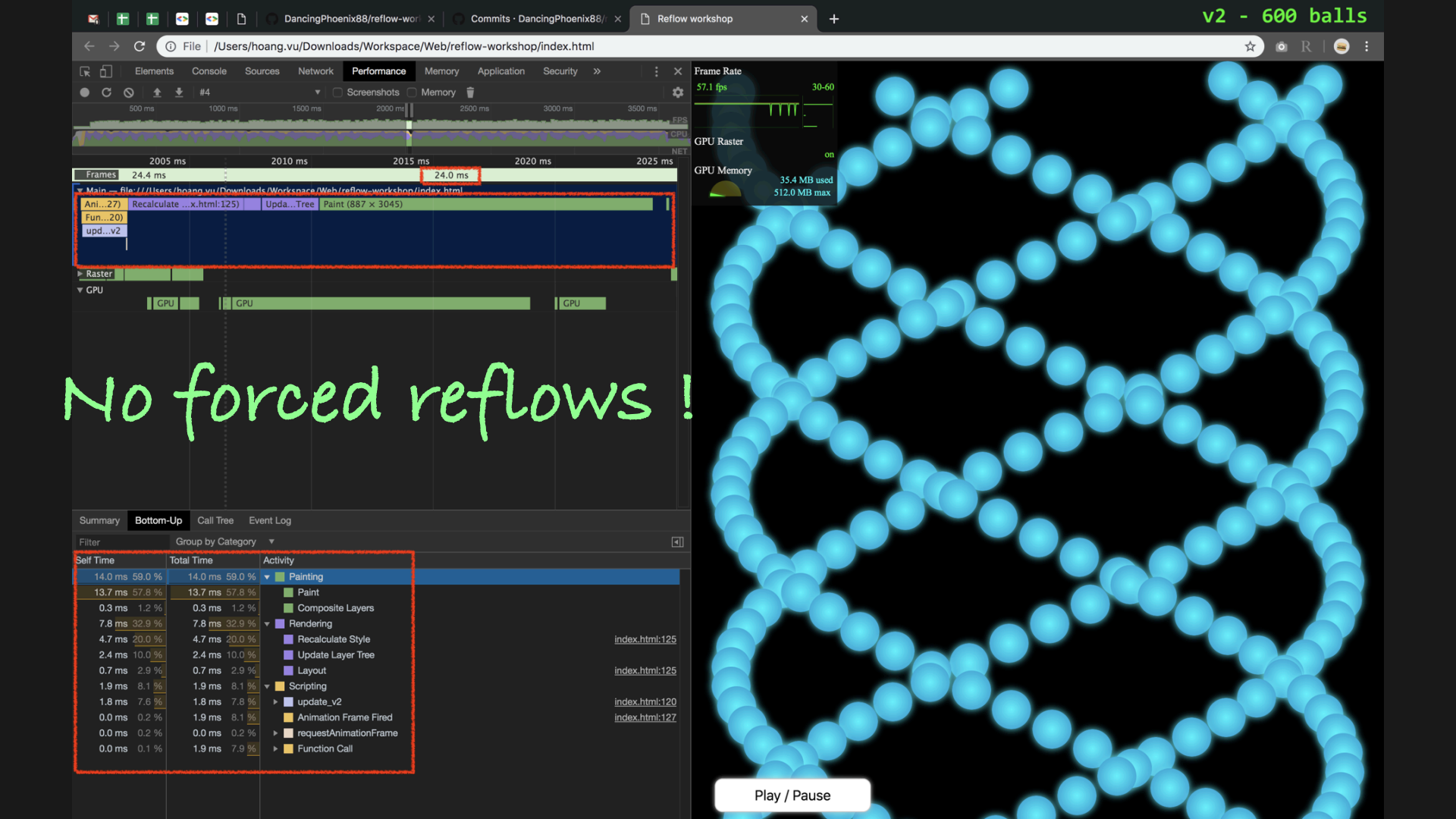
Version 2

- Branch: feature/animate-position/v2
- Key technique: Separate getting & setting style

```
function update_v2 (timestamp) {
  for (var m = 0; m < MOVERS_COUNT; m++) {
    lefts[m] = topToLeft(movers[m].offsetTop, timestamp);
  }
  for (var m = 0; m < MOVERS_COUNT; m++) {
    movers[m].style.left = lefts[m] + 'px';
  }
  if (window.running) rAF( updateFunc );
}
```

Get layout attribute

Set layout attribute


v2 - 600 balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
Frames 24.4 ms 24.0 ms
Recalculate ...x.html:125)
Paint (887 × 3045)
Raster
GPU
Frame Rate
57.1 fps 30-60
GPU Raster on
GPU Memory 35.4 MB used 512.0 MB max
No forced reflows !
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
14.0 ms 59.0 % 14.0 ms 59.0 % Painting
13.7 ms 57.8 % 13.7 ms 57.8 % Paint
0.3 ms 1.2 % 0.3 ms 1.2 % Composite Layers
7.8 ms 32.9 % 7.8 ms 32.9 % Rendering
4.7 ms 20.0 % 4.7 ms 20.0 % Recalculate Style index.html:125
2.4 ms 10.0 % 2.4 ms 10.0 % Update Layer Tree
0.7 ms 2.9 % 0.7 ms 2.9 % Layout index.html:125
1.9 ms 8.1 % 1.9 ms 8.1 % Scripting
1.8 ms 7.6 % 1.8 ms 7.8 % update_v2 index.html:120
0.0 ms 0.2 % 1.9 ms 8.1 % Animation Frame Fired index.html:127
0.0 ms 0.2 % 0.0 ms 0.2 % requestAnimationFrame
0.0 ms 0.1 % 1.9 ms 7.9 % Function Call
Play / Pause

Version 2

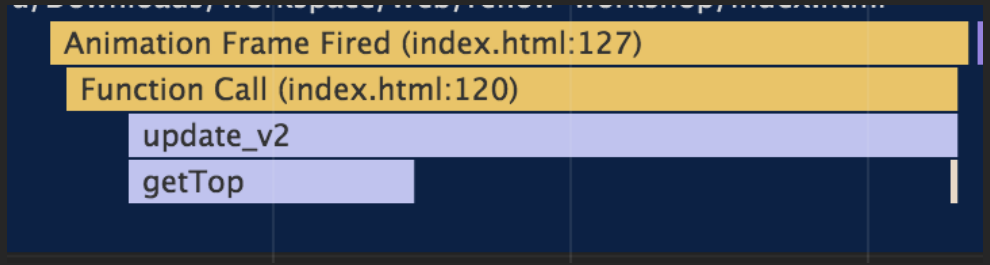
- Branch: feature/animate-position/v2
- Key technique: Separate getting & setting style
- Good:
  - No forced reflows
  - Animate 600 balls @ 60FPS

600 balls

Version 2

- Branch: [feature/animate-position/v2]
- Key technique: Separate getting & setting style
- Just getting '`offsetTop`' takes 20-30% time. And they are FIXED numbers !!!

```
function getTop(m) { return movers[m].offsetTop; }
```

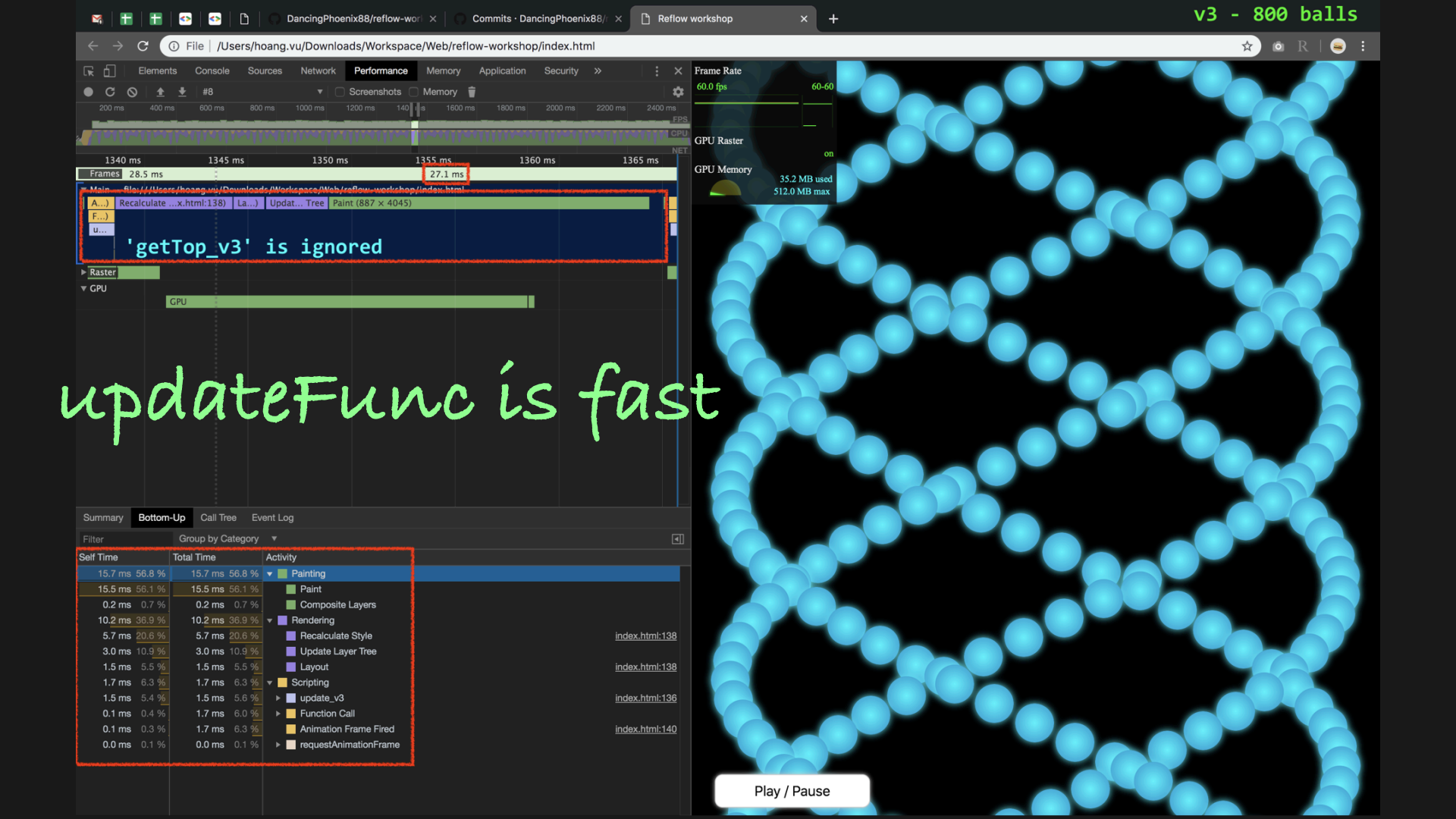


Version 3

- Branch: feature/animate-position/v3
- Key technique: Cache frequently-used values

```
function update_v3 (timestamp) {
  for (var m = 0; m < MOVERS_COUNT; m++) {
    movers[m].style.left = topToLeft(tops[m], timestamp) + 'px';
  }
  if (window.running) rAF( updateFunc );
}
```

Store 'offsetTop' in 'tops' array


v3 - 800 balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
Frames 28.5 ms 27.1 ms
'getTop_v3' is ignored
Raster
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
15.7 ms 56.8 % 15.7 ms 56.8 % Painting
15.5 ms 56.1 % 15.5 ms 56.1 % Paint
0.2 ms 0.7 % 0.2 ms 0.7 % Composite Layers
10.2 ms 36.9 % 10.2 ms 36.9 % Rendering
5.7 ms 20.6 % 5.7 ms 20.6 % Recalculate Style index.html:138
3.0 ms 10.9 % 3.0 ms 10.9 % Update Layer Tree
1.5 ms 5.5 % 1.5 ms 5.5 % Layout index.html:138
1.7 ms 6.3 % 1.7 ms 6.3 % Scripting
1.5 ms 5.4 % 1.5 ms 5.6 % update_v3 index.html:136
0.1 ms 0.4 % 1.7 ms 6.0 % Function Call
0.1 ms 0.3 % 1.7 ms 6.3 % Animation Frame Fired index.html:140
0.0 ms 0.1 % 0.0 ms 0.1 % requestAnimationFrame
Frame Rate
60.0 fps 60-60
GPU Raster on
GPU Memory 35.2 MB used 512.0 MB max
updateFunc is fast
Play / Pause

800 balls

Version 3

- Branch: feature/animate-position/v3
- Key technique: Cache frequently-used values
- Good: Can handle 800 balls
- Bad:
  - Rendering (calculating style & layout) takes too much time (10.2ms)
  - Painting takes too much time (15.7ms)
  - *Budget: 16.7ms in total to get 60 FPS

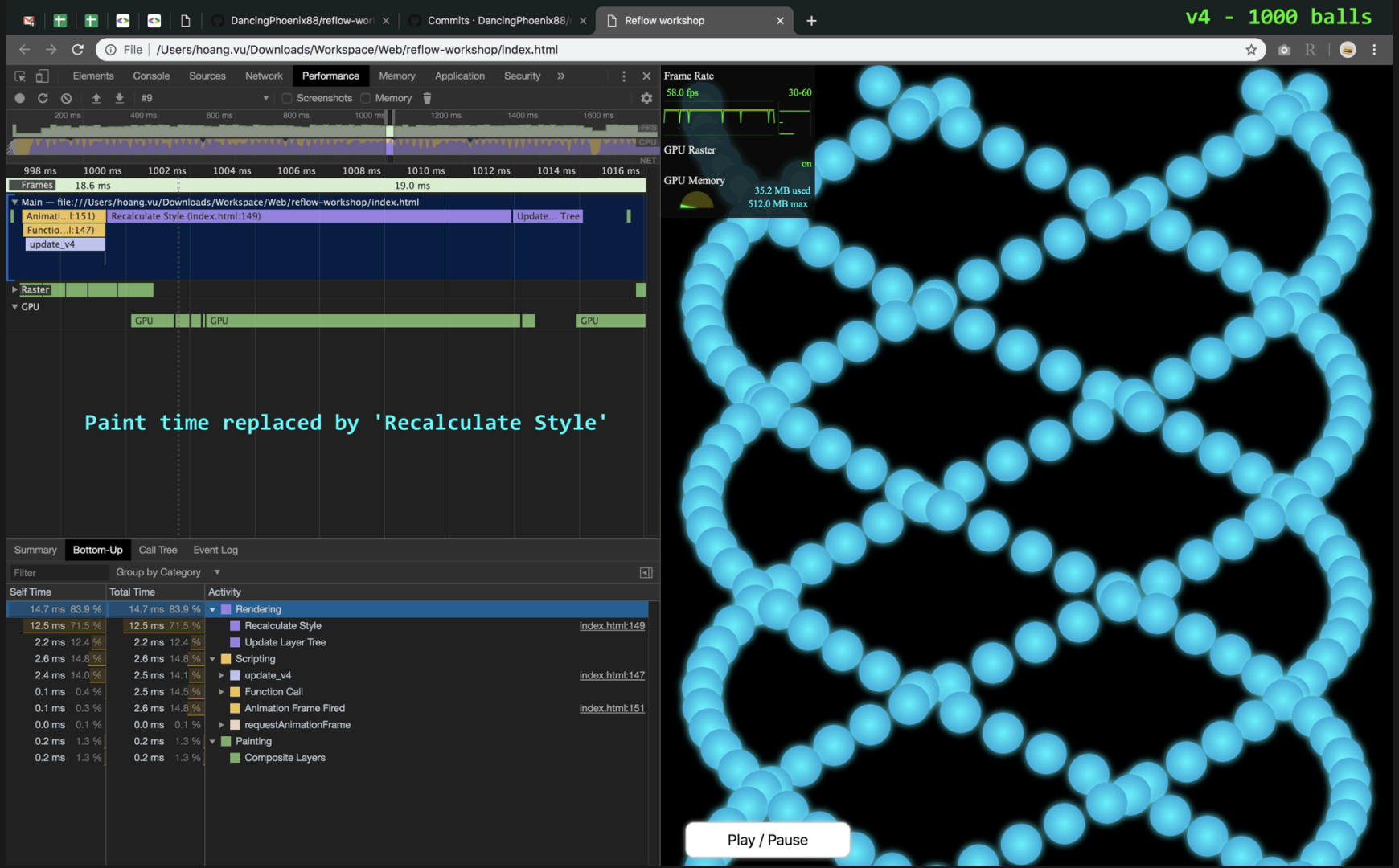
top & left cause reflow,

transform doesn’t

Version 4

- Branch: [feature/animate-position/v4]
- Key technique: Replace (‘`top`’ & ‘`left`’) by ‘`transform(translate)`’

```
function update_v4 (timestamp) {
  for (var m = 0; m < MOVERS_COUNT; m++) {
    movers[m].style.transform = `translate(${topToLeft(tops[m], timestamp)}px, ${tops[m]}px)`;
  }
  if (window.running) rAF( updateFunc );
}
```


v4 - 1000 balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
Frames 18.6 ms 19.0 ms
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Animati...l:151) Recalculate Style (index.html:149) Update... Tree
Functio...l:147)
update_v4
Raster
GPU
Paint time replaced by 'Recalculate Style'
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
14.7 ms 83.9 % 14.7 ms 83.9 % Rendering
12.5 ms 71.5 % 12.5 ms 71.5 % Recalculate Style index.html:149
2.2 ms 12.4 % 2.2 ms 12.4 % Update Layer Tree
2.6 ms 14.8 % 2.6 ms 14.8 % Scripting
2.4 ms 14.0 % 2.5 ms 14.1 % update_v4 index.html:147
0.1 ms 0.4 % 2.5 ms 14.5 % Function Call
0.1 ms 0.3 % 2.6 ms 14.8 % Animation Frame Fired index.html:151
0.0 ms 0.1 % 0.0 ms 0.1 % requestAnimationFrame
0.2 ms 1.3 % 0.2 ms 1.3 % Painting
0.2 ms 1.3 % 0.2 ms 1.3 % Composite Layers
Frame Rate
58.0 fps 30-60
GPU Raster on
GPU Memory 35.2 MB used 512.0 MB max
Play / Pause


1,000 balls

Version 4

- Branch: feature/animate-position/v4
- Key technique: Replace (‘`top`’ & ‘`left`’) by ‘`transform(translate)`’
- Good:
  - Can handle 1000 balls (not so much improved)
  - No re-paint
- Bad: Recalculating style still takes too much time

Version 3

- Re-calculate ‘left’ only
- Restyle & Reflow takes 33% time
- Re-paint takes 60% time
- 800 balls

I can deal with this
(version 5)

Version 4

- Calculate ‘x’ & ‘y’ in ‘translate’
- Restyle & Reflow takes 84% time
- No re-paint
- 1000 balls

No idea to reduce yet
But no-repaint is attractive

Where are 1,000 balls ?

Oh, they’re not in viewport

Why draw balls we can’t see ???

Version 5

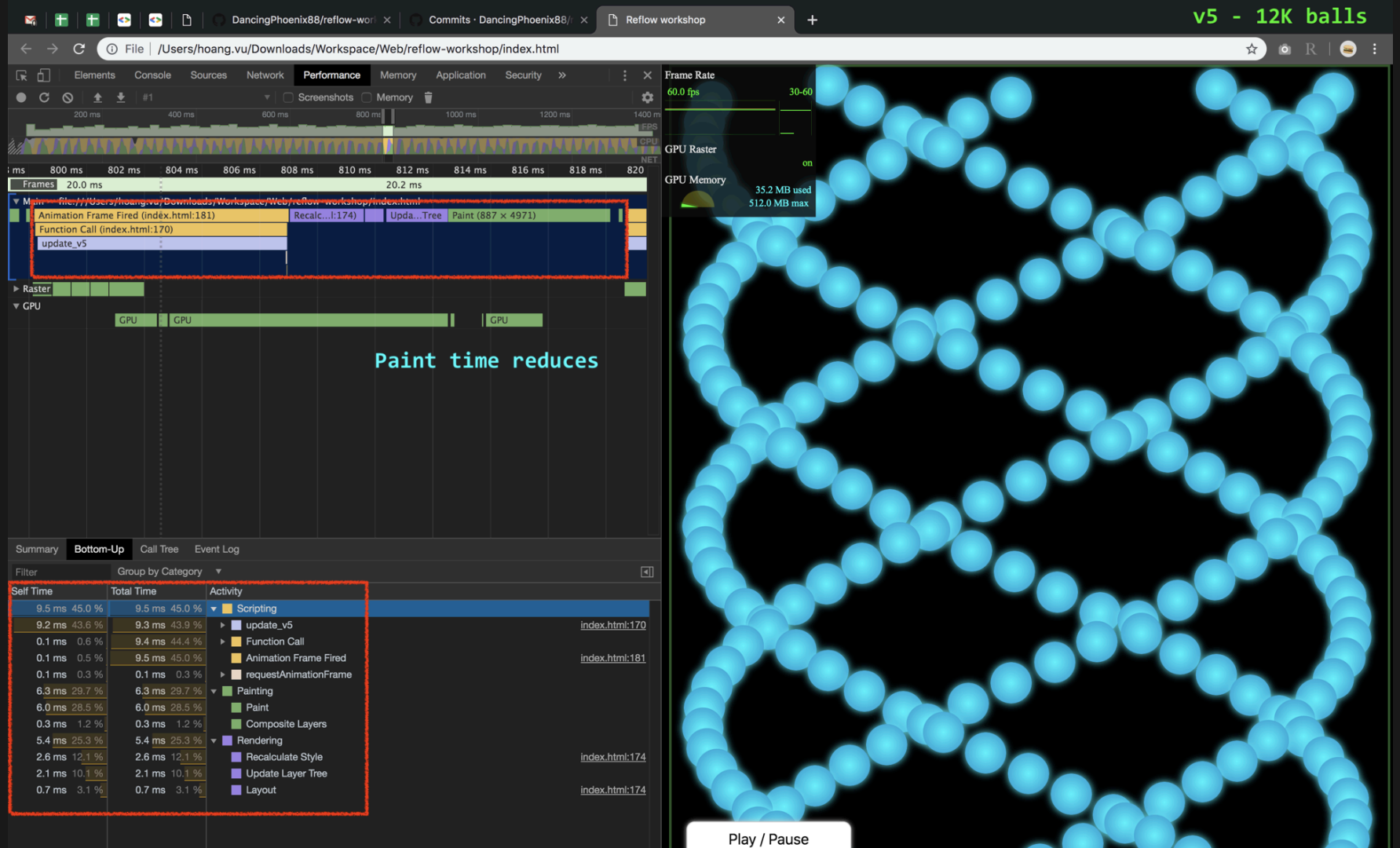
- Branch: feature/animate-position/v5
- Key technique: Frustum Culling (hide elements not in viewport / camera)

Version 5

```
function update_v5 (timestamp) {
  var scrollY = window.scrollY;
  for (var m = 0; m < MOVERS_COUNT; m++) {
    if (isInViewport(tops[m], scrollY)) {
      movers[m].style.left    = topToLeft(tops[m], timestamp) + 'px';
      movers[m].style.display = 'block';
    } else if (m != MOVERS_COUNT - 1) { // never hide the last mover to keep page size
      movers[m].style.display = 'none';
    }
  }
  if (window.running) rAF( updateFunc );
}
```

Try to update style when DOM node is invisible


12,000 balls


v5 - 12K balls
DancingPhoenix88/reflow-wor
Commits · DancingPhoenix88/
Reflow workshop
File | /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Screenshots
Memory
Frames
20.0 ms
20.2 ms
Animation Frame Fired (index.html:181)
Recalc...l:174)
Upda...Tree
Paint (887 × 4971)
Function Call (index.html:170)
update_v5
Raster
GPU
Paint time reduces
Summary
Bottom-Up
Call Tree
Event Log
Filter
Group by Category
Self Time
Total Time
Activity
9.5 ms 45.0 %
Scripting
9.2 ms 43.6 %
update_v5
index.html:170
0.1 ms 0.6 %
Function Call
0.1 ms 0.5 %
Animation Frame Fired
index.html:181
0.1 ms 0.3 %
requestAnimationFrame
6.3 ms 29.7 %
Painting
6.0 ms 28.5 %
Paint
0.3 ms 1.2 %
Composite Layers
5.4 ms 25.3 %
Rendering
2.6 ms 12.1 %
Recalculate Style
index.html:174
2.1 ms 10.1 %
Update Layer Tree
0.7 ms 3.1 %
Layout
index.html:174
Frame Rate
60.0 fps
30-60
GPU Raster
on
GPU Memory
35.2 MB used
512.0 MB max
Play / Pause

Chrome Devtools

Tab ‘Layer’

Version 5

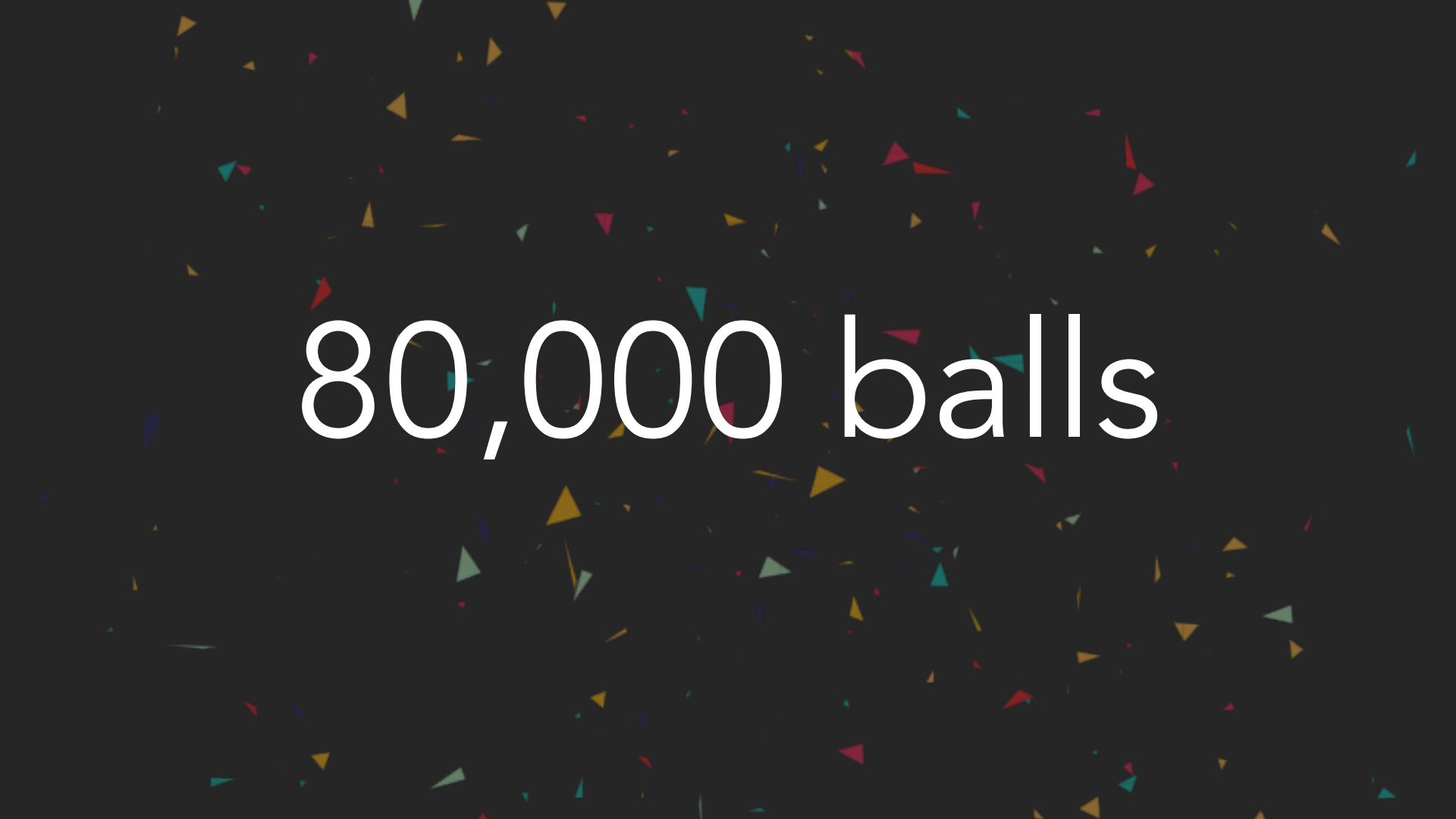
- Branch: feature/animate-position/v5
- Key technique: Frustum Culling (hide elements not in viewport / camera)
- Good:
  - Can handle 12K balls
  - Painting time is mostly fixed no matter how many balls we have
- Bad: '`display:none`' causes reflow

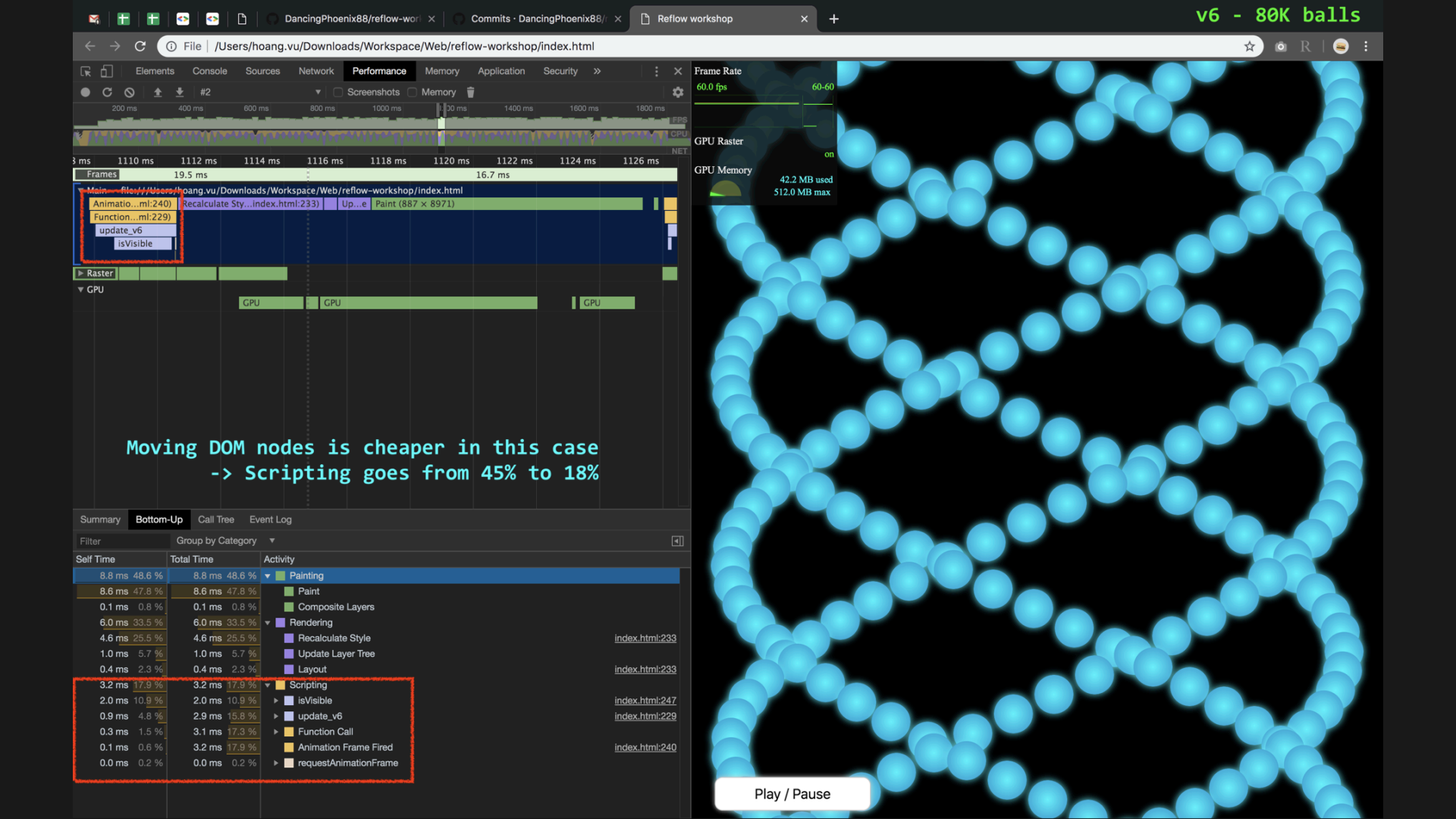
Version 6

- Branch: feature/animate-position/v6
- Key technique: Hiding elements by moving between parent nodes

Version 6

```
function update_v6 (timestamp) {
  var scrollY = window.scrollY;
  for (var m = 0; m < MOVERS_COUNT; m++) {
    if (isInViewport(tops[m], scrollY)) {
      movers[m].style.left = topToLeft(tops[m], timestamp) + 'px';
      if (isHidden(m)) showImmediately(m);
    } else if (m != MOVERS_COUNT - 1) { // never hide the last mover to keep page size
      if (isVisible(m)) hide(m);
    }
  }
  if (window.running) rAF( updateFunc );
}
```


80,000 balls


v6 - 80K balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
Frames 19.5 ms 16.7 ms
Animatio...ml:240)
Recalculate Sty...index.html:233)
Paint (887 × 8971)
Function...ml:229)
update_v6
isVisible
Raster
GPU
Moving DOM nodes is cheaper in this case
-> Scripting goes from 45% to 18%
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
8.8 ms 48.6 % 8.8 ms 48.6 % Painting
8.6 ms 47.8 % 8.6 ms 47.8 % Paint
0.1 ms 0.8 % 0.1 ms 0.8 % Composite Layers
6.0 ms 33.5 % 6.0 ms 33.5 % Rendering
4.6 ms 25.5 % 4.6 ms 25.5 % Recalculate Style index.html:233
1.0 ms 5.7 % 1.0 ms 5.7 % Update Layer Tree
0.4 ms 2.3 % 0.4 ms 2.3 % Layout index.html:233
3.2 ms 17.9 % 3.2 ms 17.9 % Scripting
2.0 ms 10.9 % 2.0 ms 10.9 % isVisible index.html:247
0.9 ms 4.8 % 2.9 ms 15.8 % update_v6 index.html:229
0.3 ms 1.5 % 3.1 ms 17.3 % Function Call
0.1 ms 0.6 % 3.2 ms 17.9 % Animation Frame Fired index.html:240
0.0 ms 0.2 % 0.0 ms 0.2 % requestAnimationFrame
Frame Rate
60.0 fps
GPU Raster
on
GPU Memory
42.2 MB used
512.0 MB max
Play / Pause

Version 6

- Branch: feature/animate-position/v6
- Key technique: Hiding elements by moving between parent nodes
- Good:
  - Can handle 80K balls
  - Painting time is mostly fixed no matter how many balls we have
- Bad: '`isInViewport`' is slow

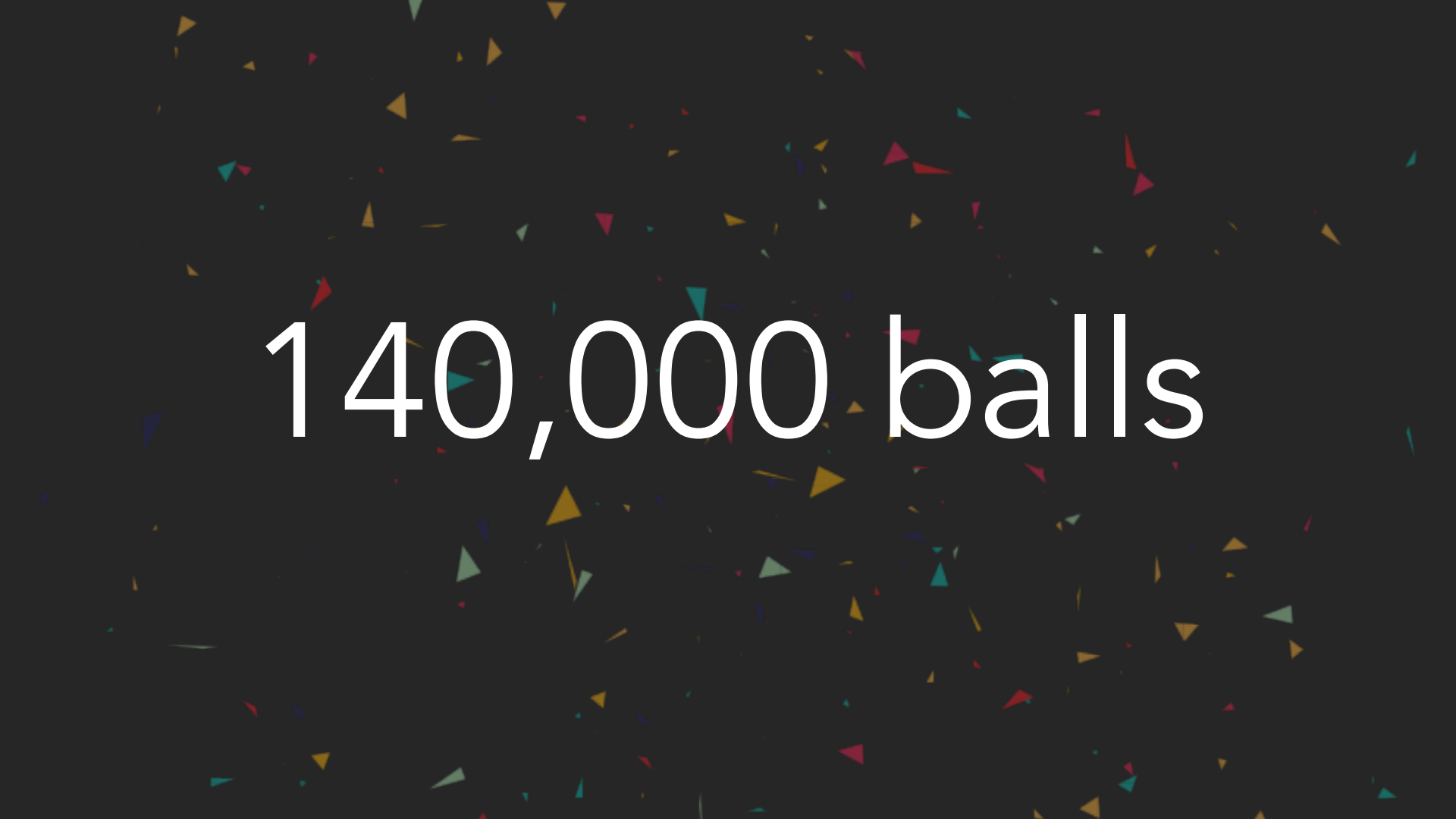
Version 6

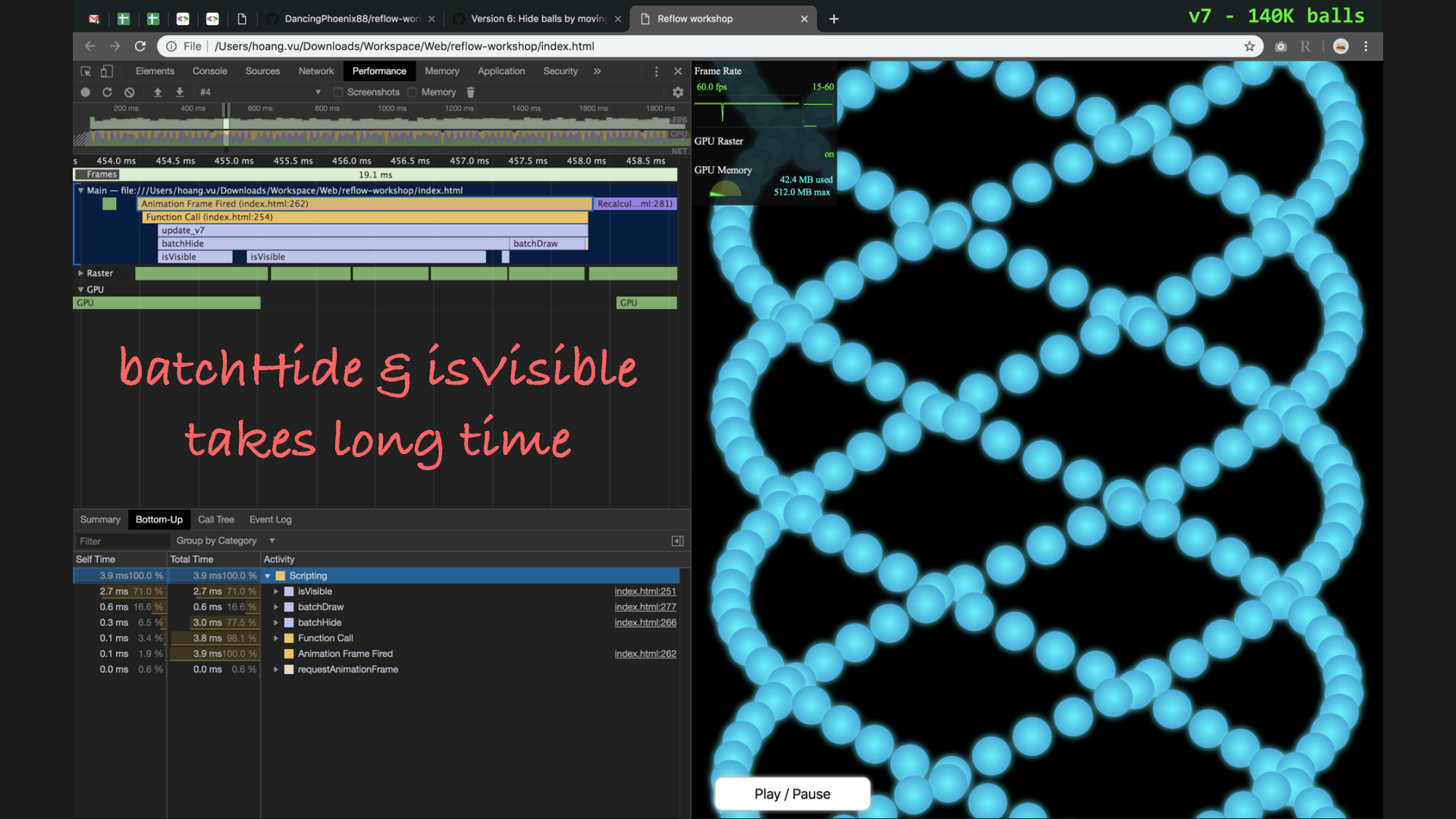
```
function update_v6 (timestamp) {
  var scrollY = window.scrollY;
  for (var m = 0; m < MOVERS_COUNT; m++) {
    if (isInViewport(tops[m], scrollY)) {
      movers[m].style.left = topToLeft(tops[m], timestamp) + 'px';
      if (isHidden(m)) showImmediately(m);
    } else if (m != MOVERS_COUNT - 1) { // never hide the last mover to keep page size
      if (isVisible(m)) hide(m);
    }
  }
  if (window.running) rAF( updateFunc );
}
```

Version 7

- Branch: feature/animate-position/v7
- Key technique: Pre-calculate visible index

```
function update_v7 (timestamp) {
  var scrollY     = window.scrollY;
  var minVisibleM = Math.floor(getMinVisibleY(scrollY) / OFFSET);
  var maxVisibleM = Math.ceil(getMaxVisibleY(scrollY) / OFFSET);
  batchHide(               0,  minVisibleM - 1             );
  batchHide( maxVisibleM + 1,     MOVERS_COUNT             ); // hide ASAP before drawing
  batchDraw(     minVisibleM,      maxVisibleM, timestamp  );
  if (window.running) rAF( updateFunc );
}
```


140,000 balls


v7 - 140K balls
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Frames 19.1 ms
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Animation Frame Fired (index.html:262)
Function Call (index.html:254)
update_v7
batchHide
batchDraw
isVisible
isVisible
Raster
GPU
batchHide & isVisible takes long time
Summary Bottom-Up Call Tree Event Log
Group by Category
Self Time Total Time Activity
3.9 ms 100.0 % 3.9 ms 100.0 % Scripting
2.7 ms 71.0 % 2.7 ms 71.0 % isVisible index.html:251
0.6 ms 16.6 % 0.6 ms 16.6 % batchDraw index.html:277
0.3 ms 6.5 % 3.0 ms 77.5 % batchHide index.html:266
0.1 ms 3.4 % 3.8 ms 98.1 % Function Call
0.1 ms 1.9 % 3.9 ms 100.0 % Animation Frame Fired index.html:262
0.0 ms 0.6 % 0.0 ms 0.6 % requestAnimationFrame
Frame Rate
60.0 fps
15-60
GPU Raster
on
GPU Memory
42.4 MB used
512.0 MB max
Play / Pause

Version 7

- Branch: [feature/animate-position/v7]
- Key technique: Pre-calculate visible index

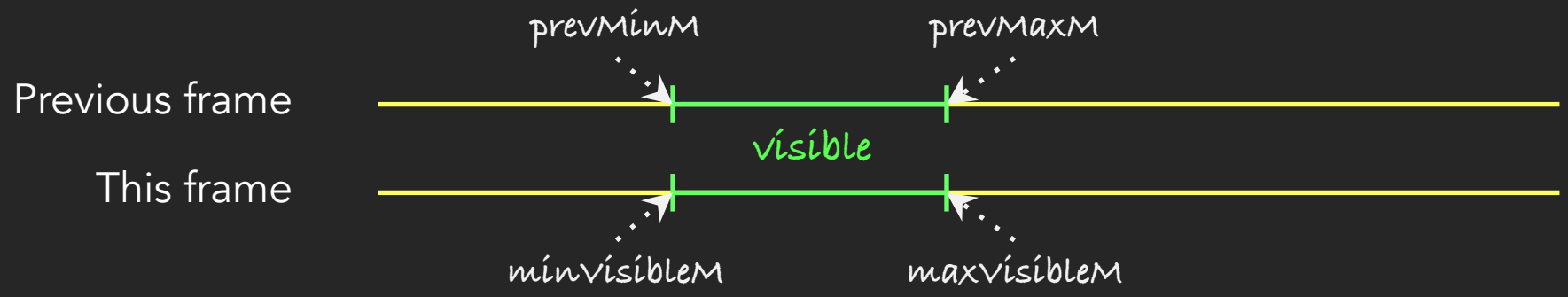
```
function update_v7 (timestamp) {
  var scrollY     = window.scrollY;
  var minVisibleM = Math.floor(getMinVisibleY(scrollY) / OFFSET);
  var maxVisibleM = Math.ceil(getMaxVisibleY(scrollY) / OFFSET);
  batchHide(               0,  minVisibleM - 1             );
  batchHide( maxVisibleM + 1,     MOVERS_COUNT             ); // hide ASAP before drawing
  batchDraw(     minVisibleM,      maxVisibleM, timestamp  );
  if (window.running) rAF( updateFunc );
}
```

Version 8

- Branch: [feature/animate-position/v8]
- Key technique: Only touch changed balls (diff)

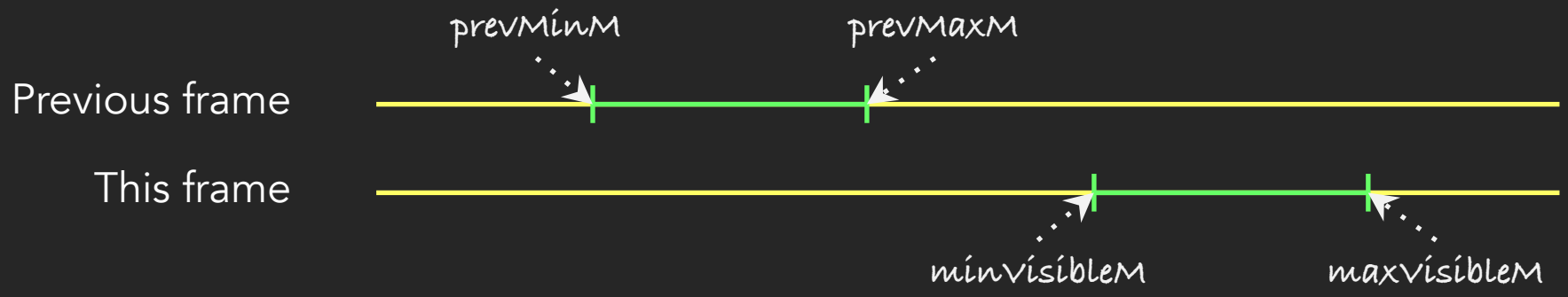
```
function update_v8 (timestamp) {
  var scrollY     = window.scrollY;
  var minVisibleM = Math.floor(getMinVisibleY(scrollY) / OFFSET);
  var maxVisibleM = Math.ceil(getMaxVisibleY(scrollY) / OFFSET);

  // Compare minVisibleM, maxVisibleM with prevMinM, prevMaxM ... (next slides)

  prevMinM = minVisibleM;
  prevMaxM = maxVisibleM;
  if (window.running) rAF( updateFunc );
}
```

Version 8 – case 1




```
if (prevMinM == minVisibleM) {
  batchMove( minVisibleM, maxVisibleM, timestamp  );
}
```

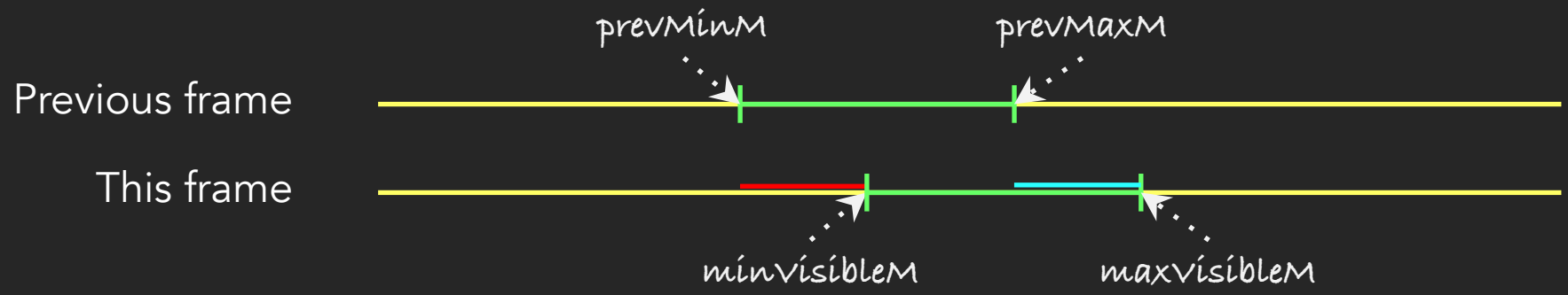
Version 8 – case 2




```
if (minVisibleM > prevMaxM || maxVisibleM < prevMinM) {
  forceHide(    prevMinM,    prevMaxM             );
  batchMove( minVisibleM, maxVisibleM, timestamp  );
  forceShow( minVisibleM, maxVisibleM             );
}
```

No need to check
slow functions: isVisible / isHidden

Version 8 – case 3 (4 is similar)



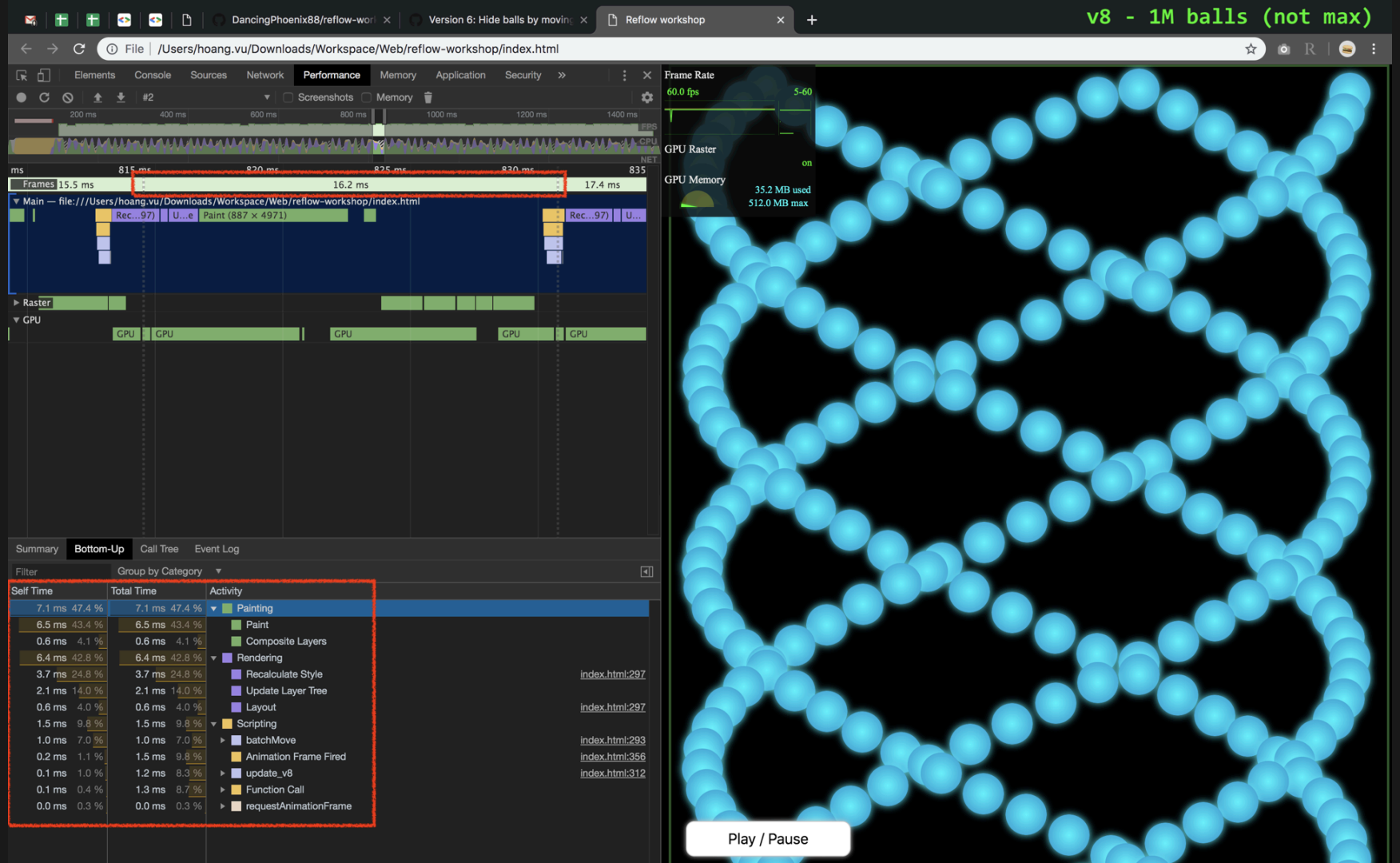

```
if (minVisibleM > prevMaxM || maxVisibleM < prevMinM) {
  forceHide(     prevMinM, minVisibleM - 1            );
  batchMove(  minVisibleM,     maxVisibleM, timestamp );
  forceShow( prevMaxM + 1,     maxVisibleM            );
}
```

Version 8 – First frame

- First frame is special
  - Hide every balls
  - Move & draw only visible balls
  - Integrated to case #2

1,000,000 balls

(not max)


v8 - 1M balls (not max)
DancingPhoenix88/reflow-wor
Version 6: Hide balls by moving
Reflow workshop
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
Frames 15.5 ms 16.2 ms 17.4 ms
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index.html
Paint (887 × 4971)
Raster
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
7.1 ms 47.4 % 7.1 ms 47.4 % Painting
6.5 ms 43.4 % 6.5 ms 43.4 % Paint
0.6 ms 4.1 % 0.6 ms 4.1 % Composite Layers
6.4 ms 42.8 % 6.4 ms 42.8 % Rendering
3.7 ms 24.8 % 3.7 ms 24.8 % Recalculate Style index.html:297
2.1 ms 14.0 % 2.1 ms 14.0 % Update Layer Tree
0.6 ms 4.0 % 0.6 ms 4.0 % Layout index.html:297
1.5 ms 9.8 % 1.5 ms 9.8 % Scripting
1.0 ms 7.0 % 1.0 ms 7.0 % batchMove index.html:293
0.2 ms 1.1 % 1.5 ms 9.8 % Animation Frame Fired index.html:356
0.1 ms 1.0 % 1.2 ms 8.3 % update_v8 index.html:312
0.1 ms 0.4 % 1.3 ms 8.7 % Function Call
0.0 ms 0.3 % 0.0 ms 0.3 % requestAnimationFrame
Frame Rate
60.0 fps 5-60
GPU Raster on
GPU Memory
35.2 MB used
512.0 MB max
Play / Pause

Version 8

- Branch: feature/animate-position/v8
- Key technique: Only touch changed balls (diff)
- Good:
  - Tested with 1M balls
  - Worst case: Process 2 x visible balls
  - → No need to improve more

WORKSHOP Part 1

Lessons:

- Use ‘`requestAnimationFrame`’
- Forced reflow is performance killer
- Grouped getting & setting styles help
- Culling reduces paint time
- Find & remove repeated calculations
- Only update changed elements
- Batch changes by DocumentFragment
- Use cloneNode to generate elements

WORKSHOP Part 1

More:

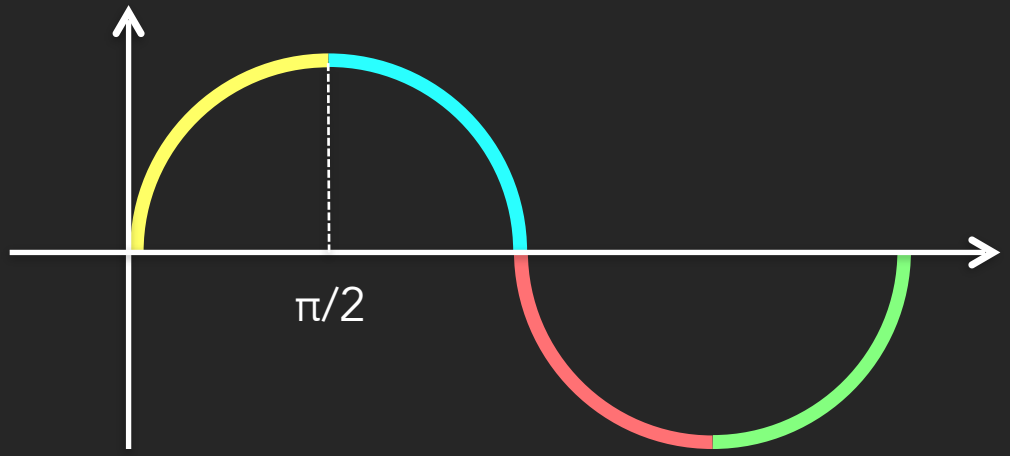
- Find & fix memory leaks
- Generate hidden balls by default
- Find which CSS attribute is heavy

PART 2

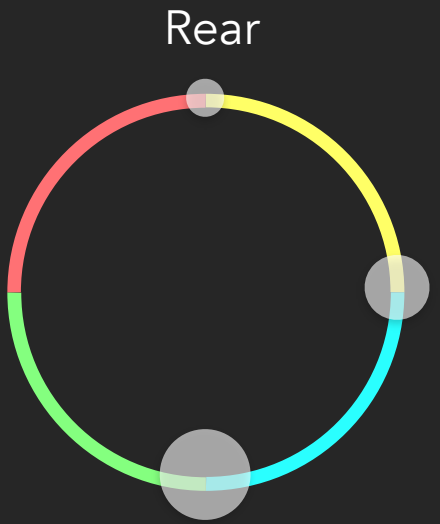
index_3d.html

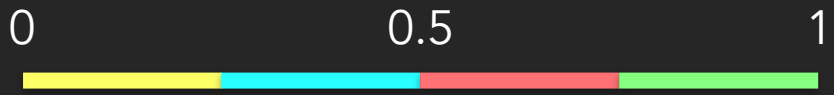
Animate as many balls as possible @ 60 FPS (in a semi-3D space)

Mimic 3D formula



sin function





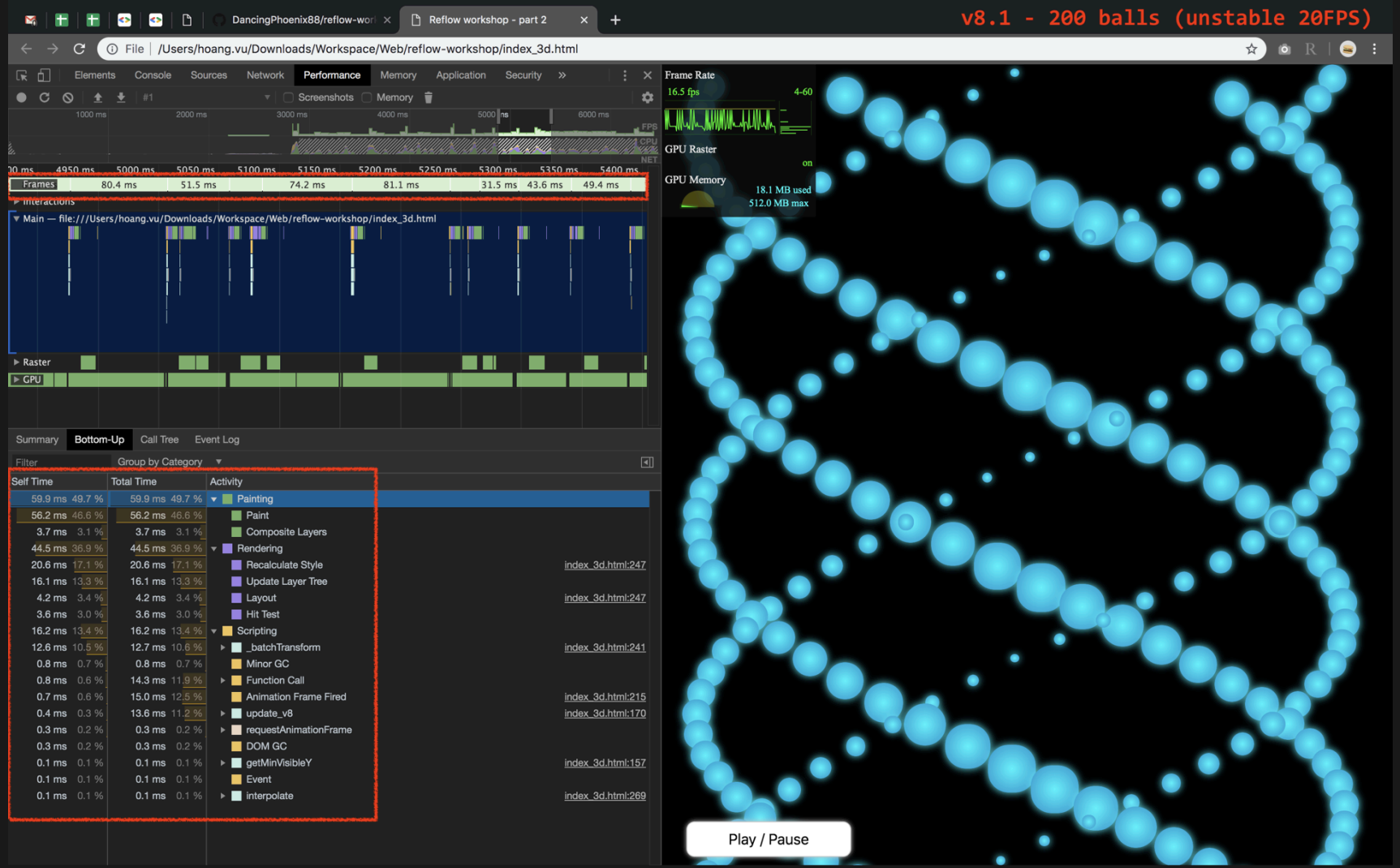
Full cycle

Version 8.1

- Branch: [feature/animate-3d/v8.1]

- Key technique: Update ‘width’ & ‘height’ over time

```
function _batchTransform(from, to, timestamp) {
  var x, left, scale;
  for (var m = from; m <= to; m++) {
    x                       = tops[m] + timestamp / MOVE_DURATION;
    left                    = (Math.sin(x) + 1) * maxLeft + minLeft;
    scale                   = cycleToScale(xToCycle(x));
    movers[m].style.left    = left + 'px';
    movers[m].style.width   =
    movers[m].style.height  = (ELEMENT_SIZE * scale) + 'px';
  }
}
```


v8.1 - 200 balls (unstable 20FPS)
DancingPhoenix88/reflow-wor
Reflow workshop - part 2
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
1000 ms 2000 ms 3000 ms 4000 ms 5000 ms 6000 ms
FPS CPU NET
4950 ms 5000 ms 5050 ms 5100 ms 5150 ms 5200 ms 5250 ms 5300 ms 5350 ms 5400 ms
Frames 80.4 ms 51.5 ms 74.2 ms 81.1 ms 31.5 ms 43.6 ms 49.4 ms
Interactions
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Raster
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
59.9 ms 49.7 % 59.9 ms 49.7 % Painting
56.2 ms 46.6 % 56.2 ms 46.6 % Paint
3.7 ms 3.1 % 3.7 ms 3.1 % Composite Layers
44.5 ms 36.9 % 44.5 ms 36.9 % Rendering
20.6 ms 17.1 % 20.6 ms 17.1 % Recalculate Style index_3d.html:247
16.1 ms 13.3 % 16.1 ms 13.3 % Update Layer Tree
4.2 ms 3.4 % 4.2 ms 3.4 % Layout index_3d.html:247
3.6 ms 3.0 % 3.6 ms 3.0 % Hit Test
16.2 ms 13.4 % 16.2 ms 13.4 % Scripting
12.6 ms 10.5 % 12.7 ms 10.6 % _batchTransform index_3d.html:241
0.8 ms 0.7 % 0.8 ms 0.7 % Minor GC
0.8 ms 0.6 % 14.3 ms 11.9 % Function Call
0.7 ms 0.6 % 15.0 ms 12.5 % Animation Frame Fired index_3d.html:215
0.4 ms 0.3 % 13.6 ms 11.2 % update_v8 index_3d.html:170
0.3 ms 0.2 % 0.3 ms 0.2 % requestAnimationFrame
0.3 ms 0.2 % 0.3 ms 0.2 % DOM GC
0.1 ms 0.1 % 0.1 ms 0.1 % getMinVisibleY index_3d.html:157
0.1 ms 0.1 % 0.1 ms 0.1 % Event
0.1 ms 0.1 % 0.1 ms 0.1 % interpolate index_3d.html:269
Frame Rate
16.5 fps 4-60
GPU Raster on
GPU Memory
18.1 MB used
512.0 MB max
Play / Pause

Version 8.1

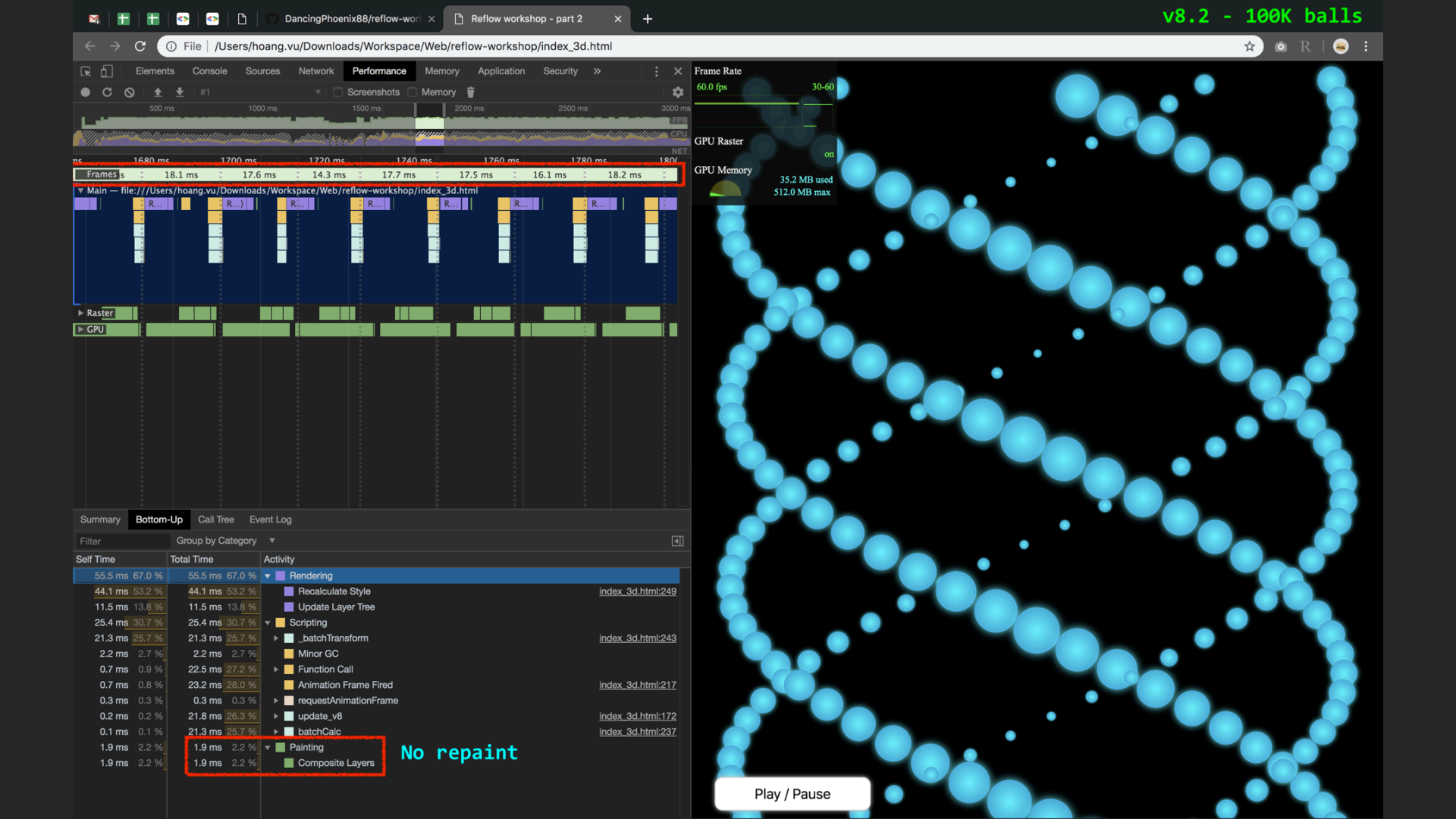
- Branch: feature/animate-3d/v8.1
- Key technique: Update '`width`' & '`height`' over time
- Problems:
  - Each ball needs to update 3 attributes (I tried '`cssText`' but still failed)
  - Many reflows
  - Frame rate is too low

Version 8.2

- Branch: feature/animate-3d/v8.2

- Key technique: Animate using ‘transform’

```
function _batchTransform(from, to, timestamp) {
  var x, left, scale;
  for (var m = from; m <= to; m++) {
    x                          = tops[m] + timestamp / MOVE_DURATION;
    left                       = (Math.sin(x) + 1) * maxLeft + minLeft;
    scale                      = cycleToScale(xToCycle(x));
    movers[m].style.transform = `translate(${left}px, ${tops[m]}px) scale(${scale})`;
  }
}
```


v8.2 - 100K balls
DancingPhoenix88/reflow-wor
Reflow workshop - part 2
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Elements Console Sources Network Performance Memory Application Security
Screenshots Memory
500 ms 1000 ms 1500 ms 2000 ms 2500 ms 3000 ms
FPS CPU NET
1680 ms 1700 ms 1720 ms 1740 ms 1760 ms 1780 ms
Frames 18.1 ms 17.6 ms 14.3 ms 17.7 ms 17.5 ms 16.1 ms 18.2 ms
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Raster
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
55.5 ms 67.0 % 55.5 ms 67.0 % Rendering
44.1 ms 53.2 % 44.1 ms 53.2 % Recalculate Style index_3d.html:249
11.5 ms 13.8 % 11.5 ms 13.8 % Update Layer Tree
25.4 ms 30.7 % 25.4 ms 30.7 % Scripting
21.3 ms 25.7 % 21.3 ms 25.7 % _batchTransform index_3d.html:243
2.2 ms 2.7 % 2.2 ms 2.7 % Minor GC
0.7 ms 0.9 % 22.5 ms 27.2 % Function Call
0.7 ms 0.8 % 23.2 ms 28.0 % Animation Frame Fired index_3d.html:217
0.3 ms 0.3 % 0.3 ms 0.3 % requestAnimationFrame
0.2 ms 0.2 % 21.8 ms 26.3 % update_v8 index_3d.html:172
0.1 ms 0.1 % 21.3 ms 25.7 % batchCalc index_3d.html:237
1.9 ms 2.2 % 1.9 ms 2.2 % Painting
1.9 ms 2.2 % 1.9 ms 2.2 % Composite Layers
No repaint
Frame Rate
60.0 fps 30-60
GPU Raster on
GPU Memory
35.2 MB used
512.0 MB max
Play / Pause

Version 8.2

- Branch: feature/animate-3d/v8.2
- Key technique: Animate using '`transform`'
- Good:
  - Can handle 100K balls @ 60FPS
  - No-repaint
- But the visual lacks of depth

Version 8.2b

- Commit: [1db447982db1dc24733cbf1c36a8db9b4a366d1d]
- Test: Add '`opacity:1`' → 100,000 balls @ 60FPS
- Test: Add '`opacity:0.8`' → 200 balls @ 30FPS !!!

If a transparent element updates,

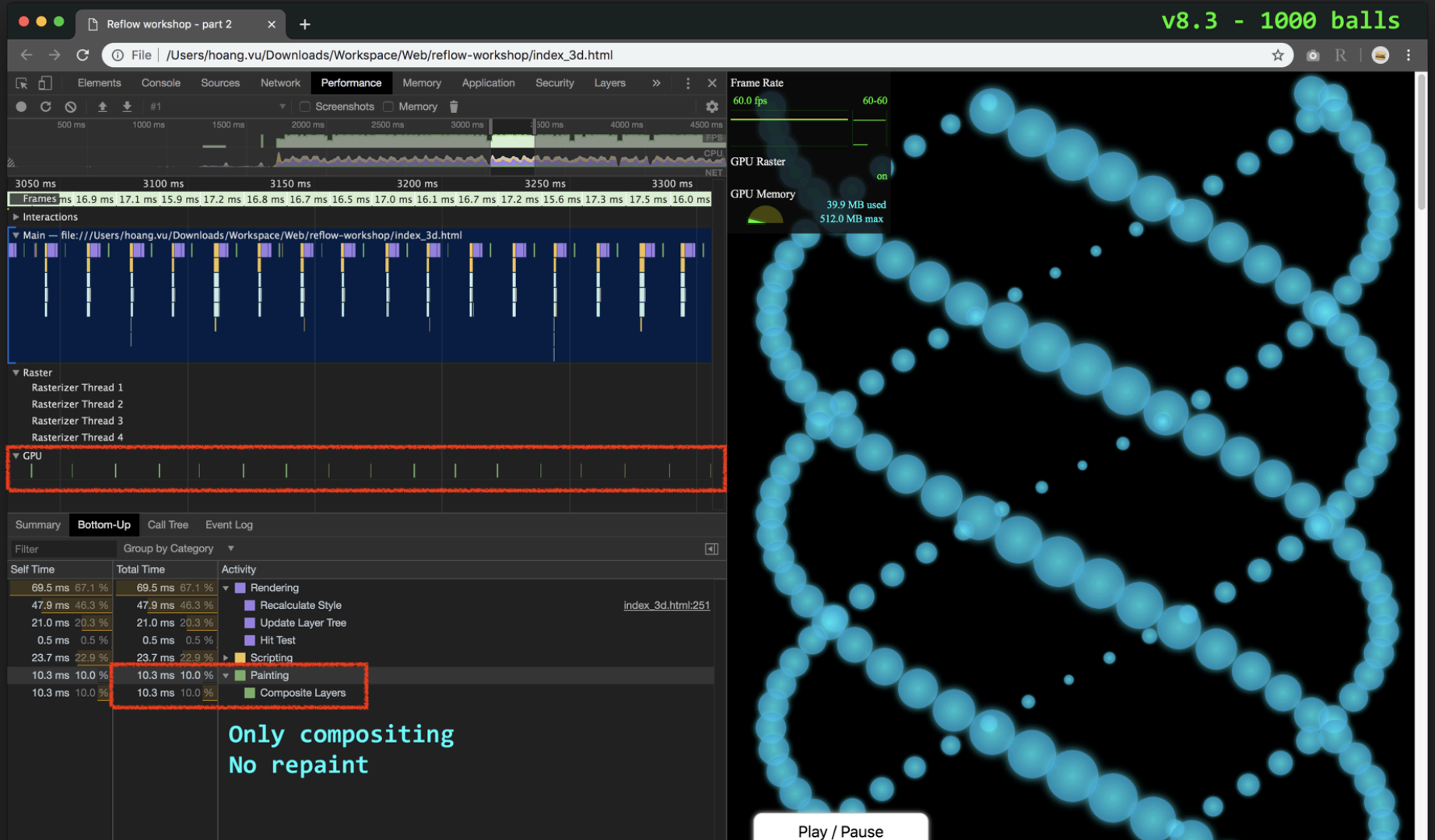
ALL overlapped ones get updating, too

Each opacity element

should be in a separate layer

Version 8.3

- Branch: feature/animate-3d/v8.3
- Key technique: Add '`will-change`' attribute
- Good:
  - Can handle 1K balls @ 60FPS
  - No-repaint
  - "Depth" added
- Visual still lacks lighting elements

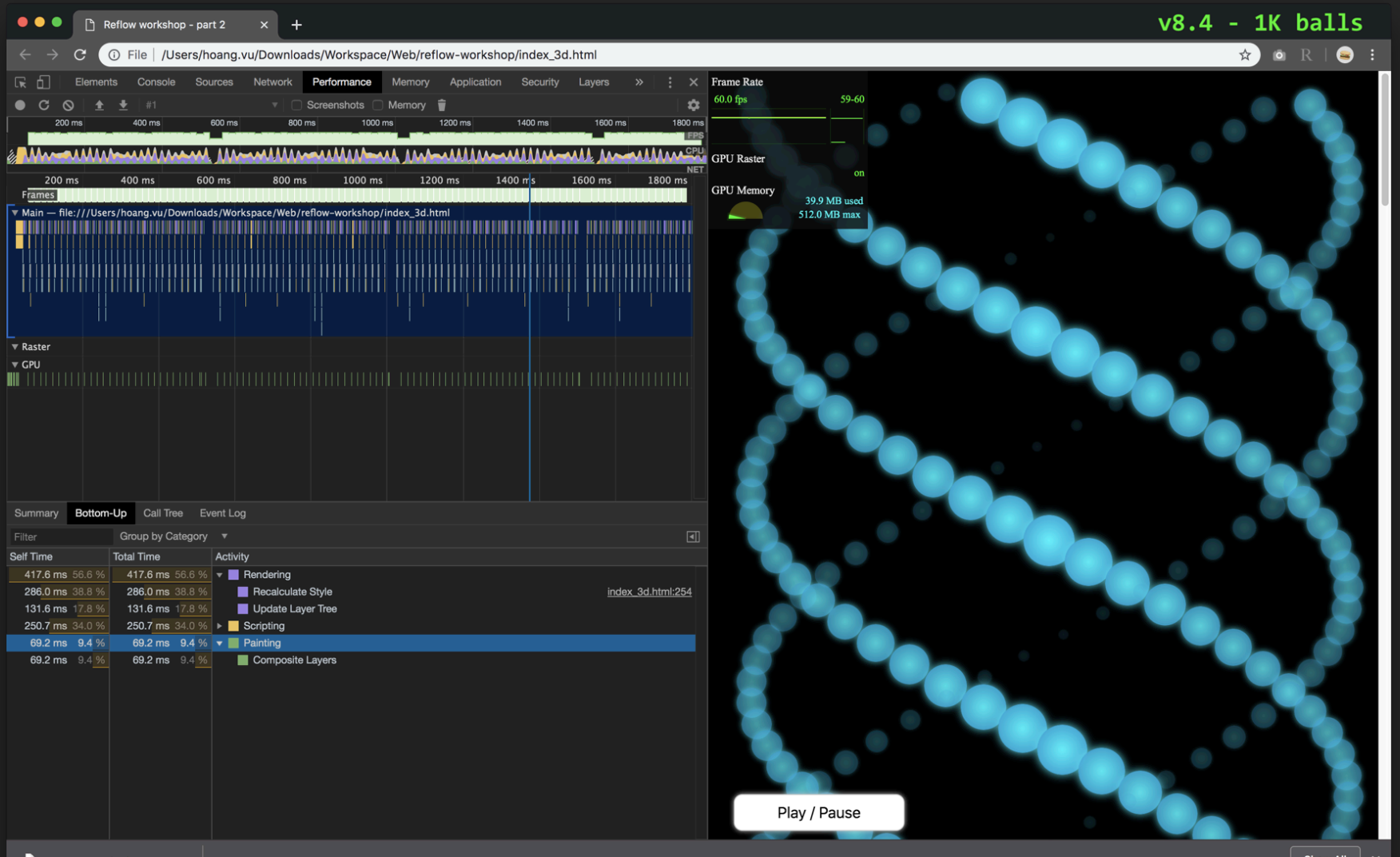

v8.3 - 1000 balls
Reflow workshop - part 2
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Elements Console Sources Network Performance Memory Application Security Layers
Screenshots Memory
Frame Rate
60.0 fps
60-60
GPU Raster
on
GPU Memory
39.9 MB used
512.0 MB max
Frames 16.9 ms 17.1 ms 15.9 ms 17.2 ms 16.8 ms 16.7 ms 16.5 ms 17.0 ms 16.1 ms 16.7 ms 17.2 ms 15.6 ms 17.3 ms 17.5 ms 16.0 ms
Interactions
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Raster
Rasterizer Thread 1
Rasterizer Thread 2
Rasterizer Thread 3
Rasterizer Thread 4
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
69.5 ms 67.1 % 69.5 ms 67.1 % Rendering
47.9 ms 46.3 % 47.9 ms 46.3 % Recalculate Style index_3d.html:251
21.0 ms 20.3 % 21.0 ms 20.3 % Update Layer Tree
0.5 ms 0.5 % 0.5 ms 0.5 % Hit Test
23.7 ms 22.9 % 23.7 ms 22.9 % Scripting
10.3 ms 10.0 % 10.3 ms 10.0 % Painting
10.3 ms 10.0 % 10.3 ms 10.0 % Composite Layers
Only compositing
No repaint
Play / Pause

Version 8.4

- Branch: [feature/animate-3d/v8.4]

- Key technique: Animate ‘opacity’ attribute

```
function _batchTransform(from, to, timestamp) {
  var x, left, scale, cycle;
  for (var m = from; m <= to; m++) {
    x                          = tops[m] + timestamp / MOVE_DURATION;
    left                       = (Math.sin(x) + 1) * maxLeft + minLeft;
    cycle                      = xToCycle(x);
    scale                      = cycleToScale(cycle);
    movers[m].style.transform = `translate(${left}px, ${tops[m]}px) scale(${scale})`;
    movers[m].style.opacity    = cycleToOpacity(cycle);
  }
}
```


Reflow workshop - part 2
v8.4 - 1K balls
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Elements Console Sources Network Performance Memory Application Security Layers
Screenshots Memory
200 ms 400 ms 600 ms 800 ms 1000 ms 1200 ms 1400 ms 1600 ms 1800 ms
FPS
CPU
NET
Frames
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d.html
Raster
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
417.6 ms 56.6 % 417.6 ms 56.6 % Rendering
286.0 ms 38.8 % 286.0 ms 38.8 % Recalculate Style index_3d.html:254
131.6 ms 17.8 % 131.6 ms 17.8 % Update Layer Tree
250.7 ms 34.0 % 250.7 ms 34.0 % Scripting
69.2 ms 9.4 % 69.2 ms 9.4 % Painting
69.2 ms 9.4 % 69.2 ms 9.4 % Composite Layers
Frame Rate
60.0 fps 59-60
GPU Raster
on
GPU Memory
39.9 MB used
512.0 MB max
Play / Pause

WORKSHOP Part 2

Lessons:

- Best CSS attributes for animation:
  - ‘`opacity`’
  - ‘`transform`’
- ‘`will-change`’ moves element to another layer

PART 3

index_3d_css.html

Animate as many balls as possible @ 60 FPS (in a semi-3D space) by CSS

Version 9.0

- Branch: [feature/animate-3d-css/v9.0]
- Key technique: CSS animation (Test with 1 ball)

```
@keyframes swing {
  0%    { opacity: 0.10; transform: translateX( 500px) scale(0.2); animation-timing-function: ease-out; }
  25%   { opacity: 0.60; transform: translateX(1000px) scale(0.7); animation-timing-function: ease-in; }
  50%   { opacity: 1.00; transform: translateX( 500px) scale(1.2); animation-timing-function: ease-out; }
  75%   { opacity: 0.60; transform: translateX(   0px) scale(0.7); animation-timing-function: ease-in; }
  100%  { opacity: 0.10; transform: translateX( 500px) scale(0.2); animation-timing-function: ease-out; }
}
.mover {
  /* ... other attributes are trimmed */
  animation-name: swing;
  animation-duration: 6.28s; /* 2 * PI * [old]MOVE_DURATION */
  animation-iteration-count: infinite;
}
```

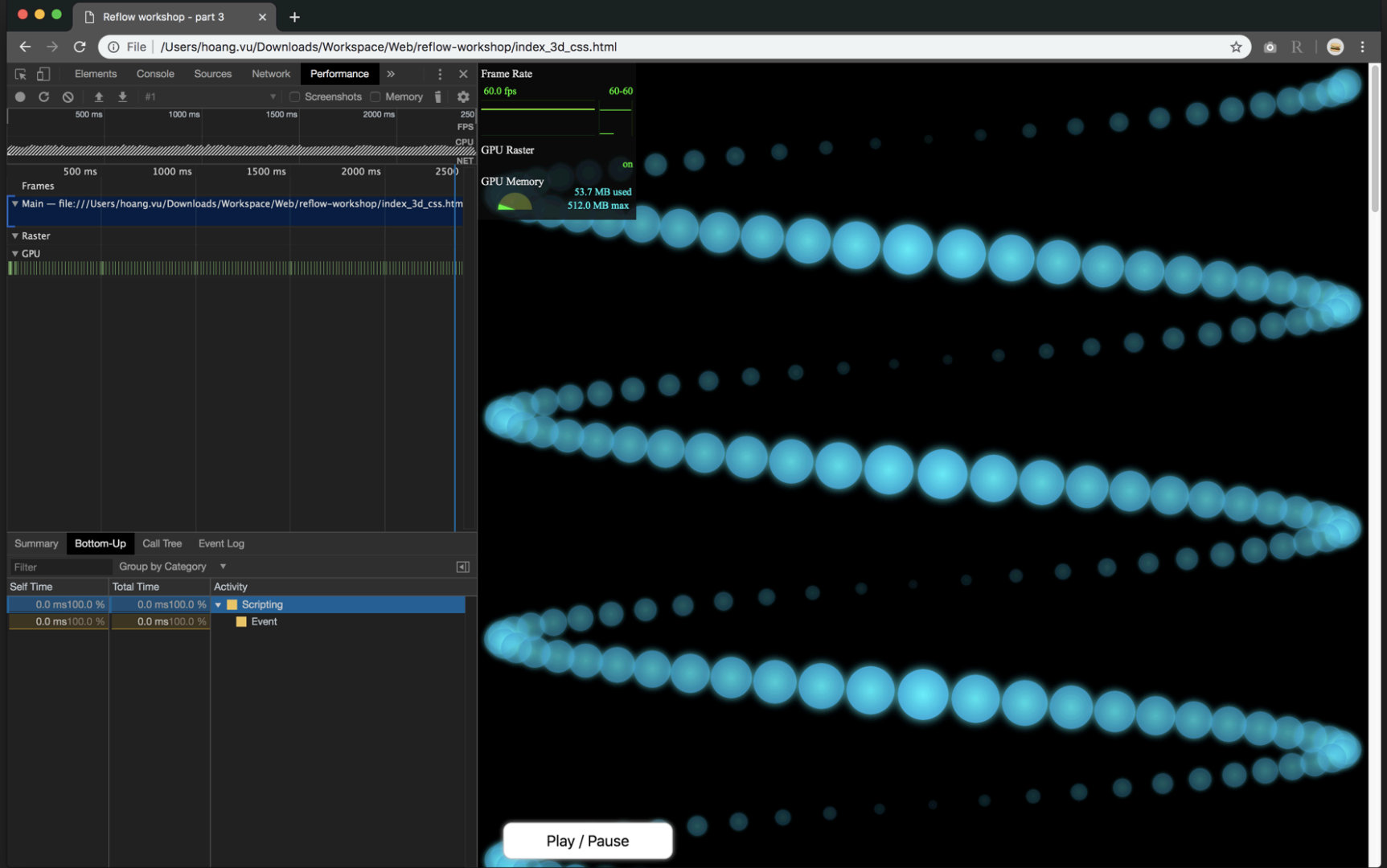
Version 9.1

- Branch: feature/animate-3d-css/v9.1
- Key technique: Just generate balls (method taken from v6)
- Balls can animate by pure CSS, but not so smooth
- Modified '`update_v8`' to show / hide elements only
- Problems: All balls start animating at the same time

Version 9.2

- Branch: [feature/animate-3d-css/v9.2]
- Key technique: Dynamic delay CSS animations
- It looks very closely to JS version
- GPU does all the job

```
var mv           = mvPrototype.cloneNode();
var delay        = (DELAY * m % ACTUAL_MOVE_DURATION) / 1000;
mv.style.cssText = `top:${m * OFFSET}px; animation-delay:${delay}s`;
```


Reflow workshop - part 3
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d_css.html
Elements
Console
Sources
Network
Performance
Screenshots
Memory
500 ms
1000 ms
1500 ms
2000 ms
FPS
CPU
NET
Frames
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d_css.htm
Raster
GPU
Summary
Bottom-Up
Call Tree
Event Log
Filter
Group by Category
Self Time
Total Time
Activity
0.0 ms 100.0 %
Scripting
Event
Frame Rate
60.0 fps
60-60
GPU Raster
on
GPU Memory
53.7 MB used
512.0 MB max
Play / Pause

But I can not pause the animations !

Version 9.3

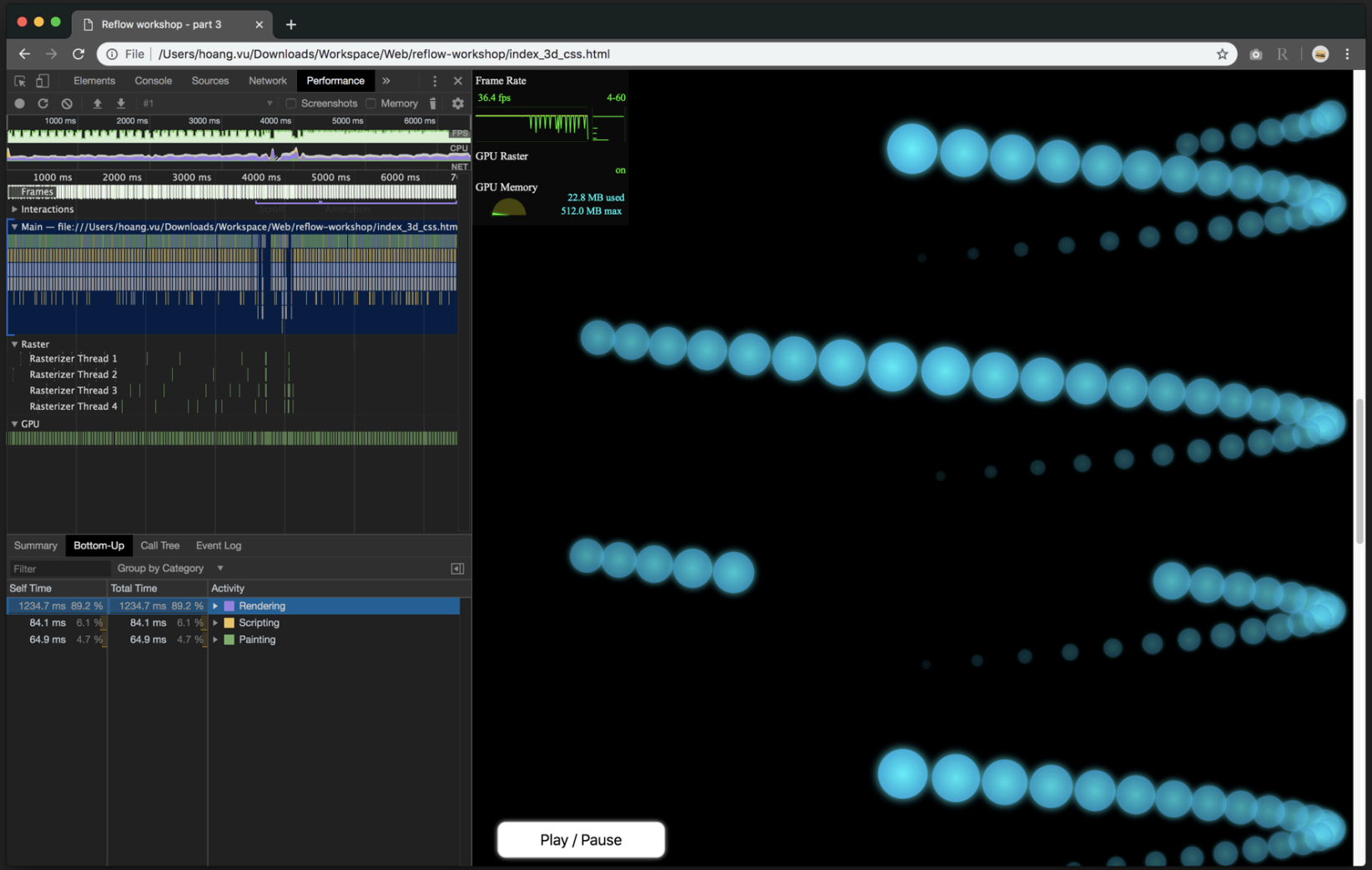
- Branch: feature/animate-3d-css/v9.3
- Key technique: Change ‘`animation-play-state`’ automatically
- Use CSS selector to control children from parent class
- Now button “Play / Pause” works normally

```
#container.pause .mover {
  animation-play-state: paused;
}
```

How about culling ?

Version 9.4

- Branch: feature/animate-3d-css/v9.4
- Key technique: Apply culling
- Problems: Out-of-sync animations


Reflow workshop - part 3
File /Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d_css.html
Elements Console Sources Network Performance
Screenshots Memory
1000 ms 2000 ms 3000 ms 4000 ms 5000 ms 6000 ms
FPS
CPU
NET
Frames
Interactions
Main — file:///Users/hoang.vu/Downloads/Workspace/Web/reflow-workshop/index_3d_css.htm
Raster
Rasterizer Thread 1
Rasterizer Thread 2
Rasterizer Thread 3
Rasterizer Thread 4
GPU
Summary Bottom-Up Call Tree Event Log
Filter Group by Category
Self Time Total Time Activity
1234.7 ms 89.2 % 1234.7 ms 89.2 % Rendering
84.1 ms 6.1 % 84.1 ms 6.1 % Scripting
64.9 ms 4.7 % 64.9 ms 4.7 % Painting
Frame Rate
36.4 fps 4-60
GPU Raster on
GPU Memory 22.8 MB used 512.0 MB max
Play / Pause

WORKSHOP
Part 3

Lessons:3

- CSS animation is useful for simple case
- CSS animation starts immediately
- It is hard to control CSS animation
- Try CSS custom var in CSS animation
  - feature/animate-3d-css/v9.5

It’s good to understand

how a web page is rendered

What forces reflow ?

- JS: https://gist.github.com/paulirish/5d52fb081b3570c81e3a
- CSS: https://csstriggers.com/

DOM & Reflow

- FastDOM (a library):
  - Queue ‘get’ & ‘set’ style commands using ‘`requestAnimationFrame`’ (like v2).
- Virtual DOM (a technique):
  - A snapshot of DOM each frame.
  - By comparing snapshots, we can send actual changes to real DOM tree (like v8).
  - Most popular with ReactJS, VueJS
- Shadow DOM (a new Web Component):
  - An isolated DOM sub-tree
  - Shadow DOM and outter nodes can’t interact with each other
  - From the outside, Shadow DOM just repaints, like v9

More references

- Presentations: https://speakerdeck.com/addyosmani/velocityconf-rendering-performance-case-studies?slide=1
- Documents hub:
  - http://jankfree.org/
  - https://developers.google.com/web/
  - http://wilsonpage.co.uk/preventing-layout-thrashing/
  - https://www.igvita.com/slides/2012/web-performance-for-the-curious/#29
  - https://www.chromium.org/developers/design-documents/gpu-accelerated-compositing-in-chrome
- Advanced profiling tools:
  - http://dev.chromium.org/developers/how-tos/trace-event-profiling-tool
  - https://www.html5rocks.com/en/tutorials/games/abouttracing/
  - https://www.gamasutra.com/view/news/176420/Indepth_Using_Chrometracing_to_view_your_inline_profiling_data.php

Q&A