Nouveaux Moyens d'Interaction

#FishEye #VR

Luc MARONGIU - 2017
Tech lead - Air France
ls.marongiu@gmail.com

POLYTECH
NICE-SOPHIA

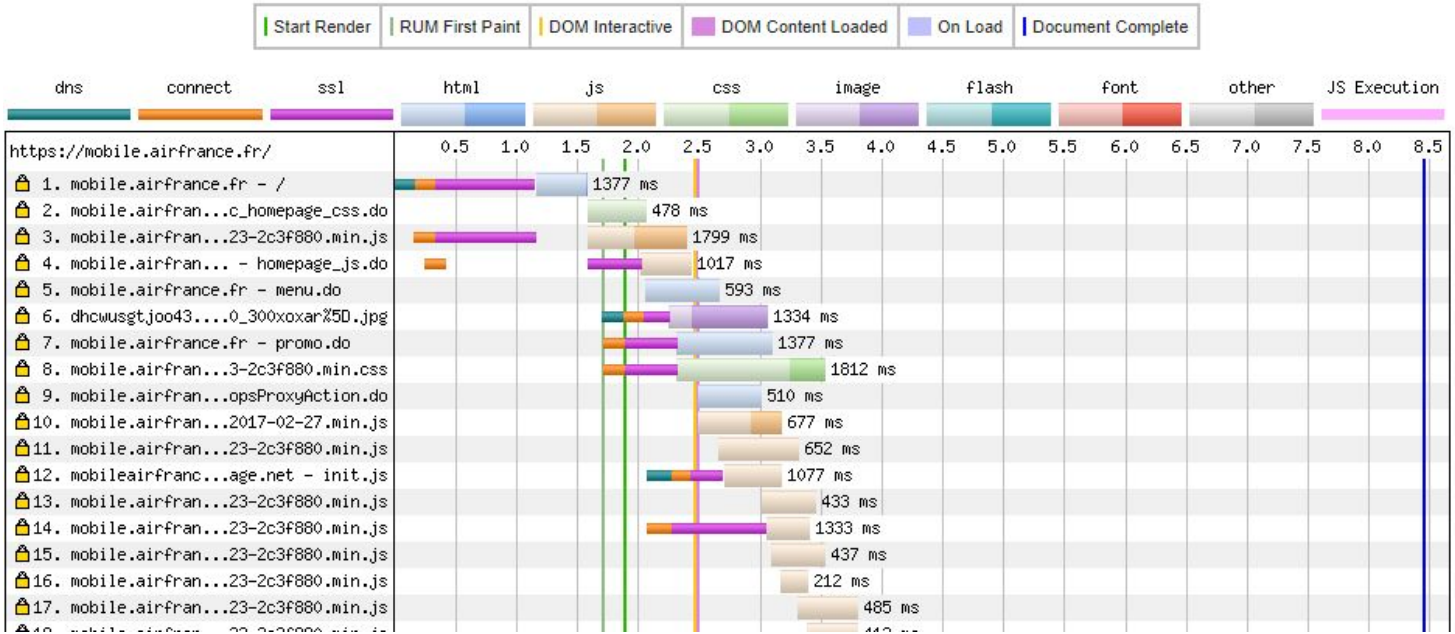
Previously…

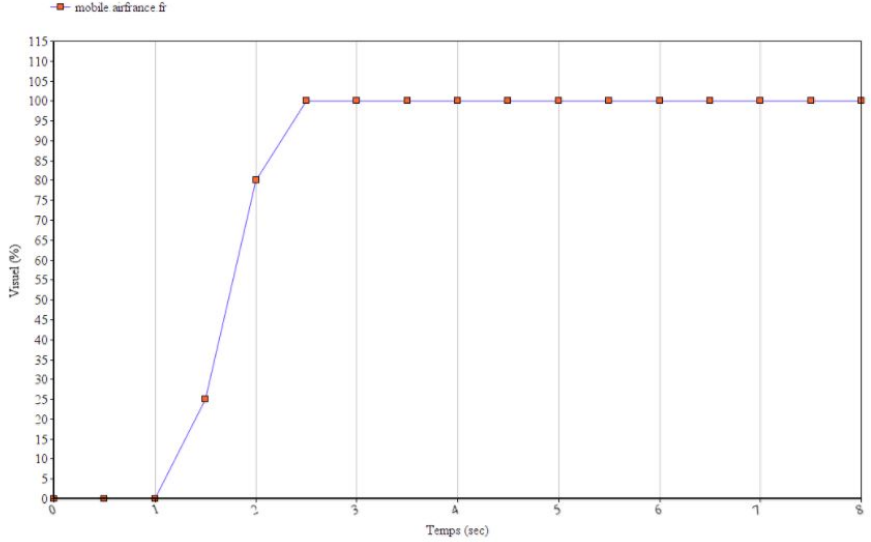
on AMC..

Load Time	First Byte	Start Render	Visually Complete	Speed Index	Result (error code)	Document Complete			Fully Loaded			
						Time	Requests	Bytes In	Time	Requests	Bytes In	Certificates
8.449s	1.567s	1.889s	3.200s	2440	0	8.449s	58	803 KB	8.642s	60	804 KB	55 KB

Colordepth	RUM First Paint	domInteractive	domContentLoaded	loadEvent
24	1.708s	2.453s	2.453s - 2.496s (0.043s)	8.439s - 8.450s (0.011s)

Waterfall View

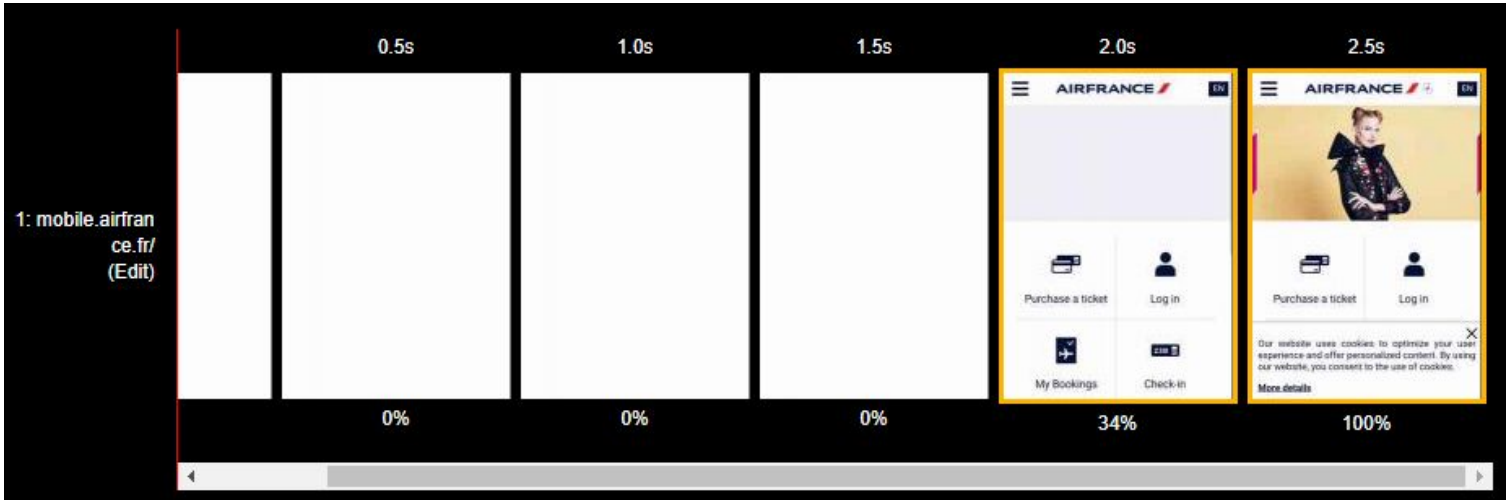




Règle n°1 : Prioriser les requêtes responsables du visuel : “*critical render path*”

Règle n°2 : Diminuer le nombre et poids de ces requêtes (inline, minification, concaténation)

Règle n°3 : La première requête ne doit pas dépasser 13.8 Ko



Visuel 100% :
2.5 sec

Load :
8.5 sec

Runtime (en ms) - http://www.stefankrause.net

Name	angular-v1.6.3-keyed	angular-v4.1.2-keyed	ember-v2.13.0	react-lite-v0.15.30	react-v15.5.4-keyed	react-v15.5.4-mobX-v3.1.9	react-v15.5.4-redux-v3.6.0	vanillajs-keyed
create rows Duration for creating 1000 rows after the page loaded.	251.88.0 (1.8)	193.17.9 (1.4)	344.613.8 (2.5)	170.09.6 (1.2)	188.910.9 (1.4)	243.99.4 (1.8)	212.214.2 (1.5)	138.55.8 (1.0)
replace all rows Duration for updating all 1000 rows of the table (with 5 warmup iterations).	278.316.7 (1.9)	197.45.3 (1.3)	292.712.1 (2.0)	235.46.8 (1.6)	201.06.4 (1.4)	229.212.2 (1.5)	206.77.3 (1.4)	148.04.5 (1.0)
swap rows Time to swap 2 rows on a 1K table. (with 5 warmup iterations).	14.71.5 (1.0)	13.41.0 (1.0)	16.41.5 (1.0)	30.01.8 (1.9)	14.70.9 (1.0)	18.01.2 (1.1)	17.11.3 (1.1)	11.41.1 (1.0)
create many rows Duration to create 10,000 rows	3108.72162.2 (2.3)	1946.041.8 (1.5)	2569.356.3 (1.9)	2300.951.4 (1.7)	1852.429.0 (1.4)	2217.371.5 (1.7)	1931.735.6 (1.5)	1331.122.2 (1.0)
append rows to large table Duration for adding 1000 rows on a table of 10,000 rows.	454.842.1 (1.5)	324.610.1 (1.1)	524.223.6 (1.8)	2087.565.2 (7.1)	345.610.4 (1.2)	459.847.2 (1.6)	366.410.9 (1.2)	295.312.8 (1.0)
clear rows Duration to clear the table filled with 10.000 rows.	817.637.2 (4.7)	379.911.3 (2.2)	303.974.7 (1.7)	344.126.4 (2.0)	398.48.2 (2.3)	495.128.8 (2.8)	410.99.8 (2.4)	174.84.2 (1.0)
startup time Time for loading, parsing and starting up	118.15.1 (2.9)	84.32.6 (2.1)	245.25.6 (6.0)	44.92.6 (1.1)	70.02.9 (1.7)	87.64.3 (2.2)	93.86.9 (2.3)	40.59.5 (1.0)
slowdown geometric mean	**2.08**	**1.45**	**2.10**	**1.92**	**1.42**	**1.74**	**1.56**	**1.00**

COÛT DEV

→ Apprendre

→ Ré-apprendre

→ Debug

COÛT UTILISATEUR

→ Temps

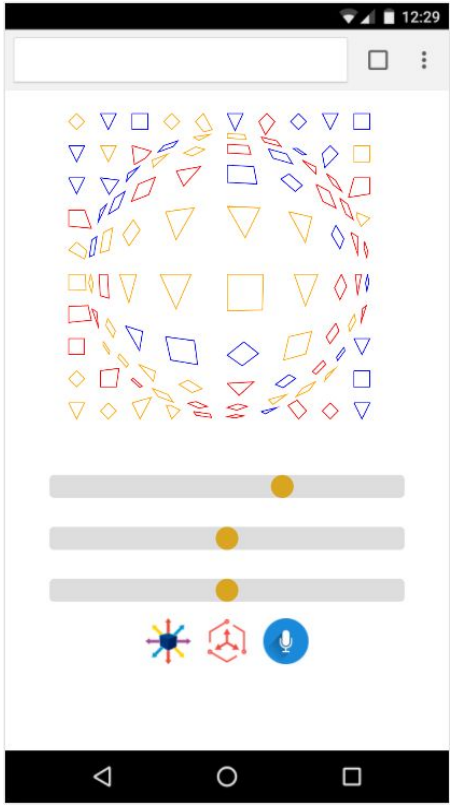
→ Bandwith

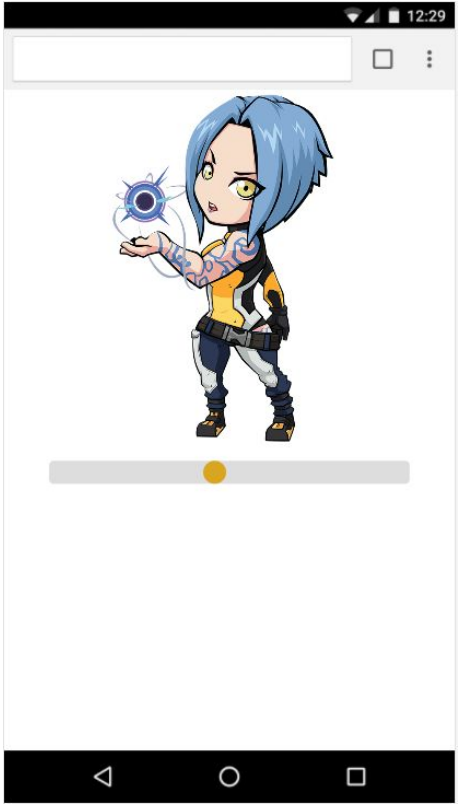
→ Utilisation CPU (batterie)

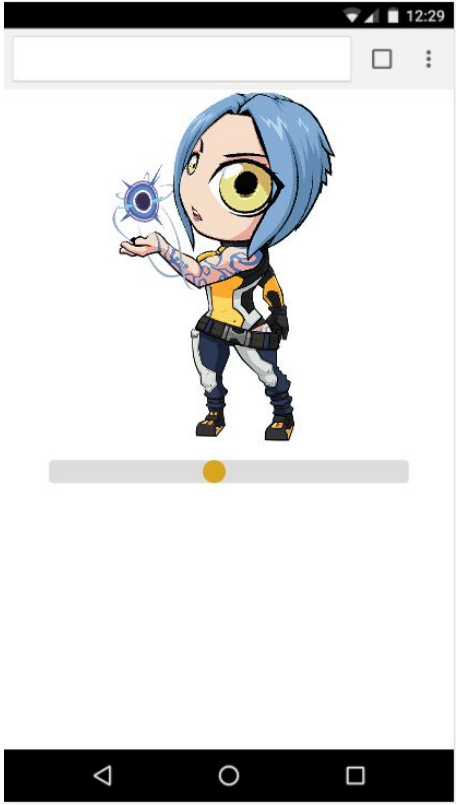
→ Frame rate

→ Utilisation mémoire

TD : Fisheye effect







NATIF

→ Performant

→ Plus de possibilité

NATIF

→ Performant

→ Plus de possibilité

WEB

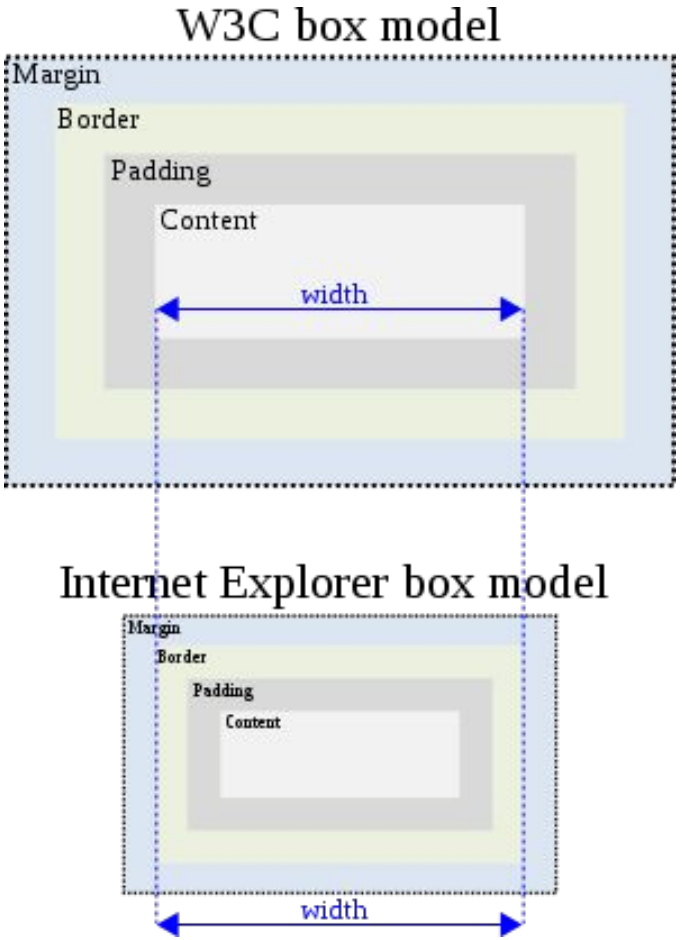
→ Accessible

→ Universel

→ Moins de code

→ Plus facile

Il fut un temps….


W3C box model
Margin
Border
Padding
Content
width
Internet Explorer box model
Margin
Border
Padding
Content
width

```
currentBtn.attachEvent("onmouseover", showHint);
```

This is the method that adds the event handler in Internet Explorer.

This time you keep the "on" at the beginning of the event name...

You still give the function the name of the handler to run when the event occurs.

That mysterious "false" disappears in attachEvent().

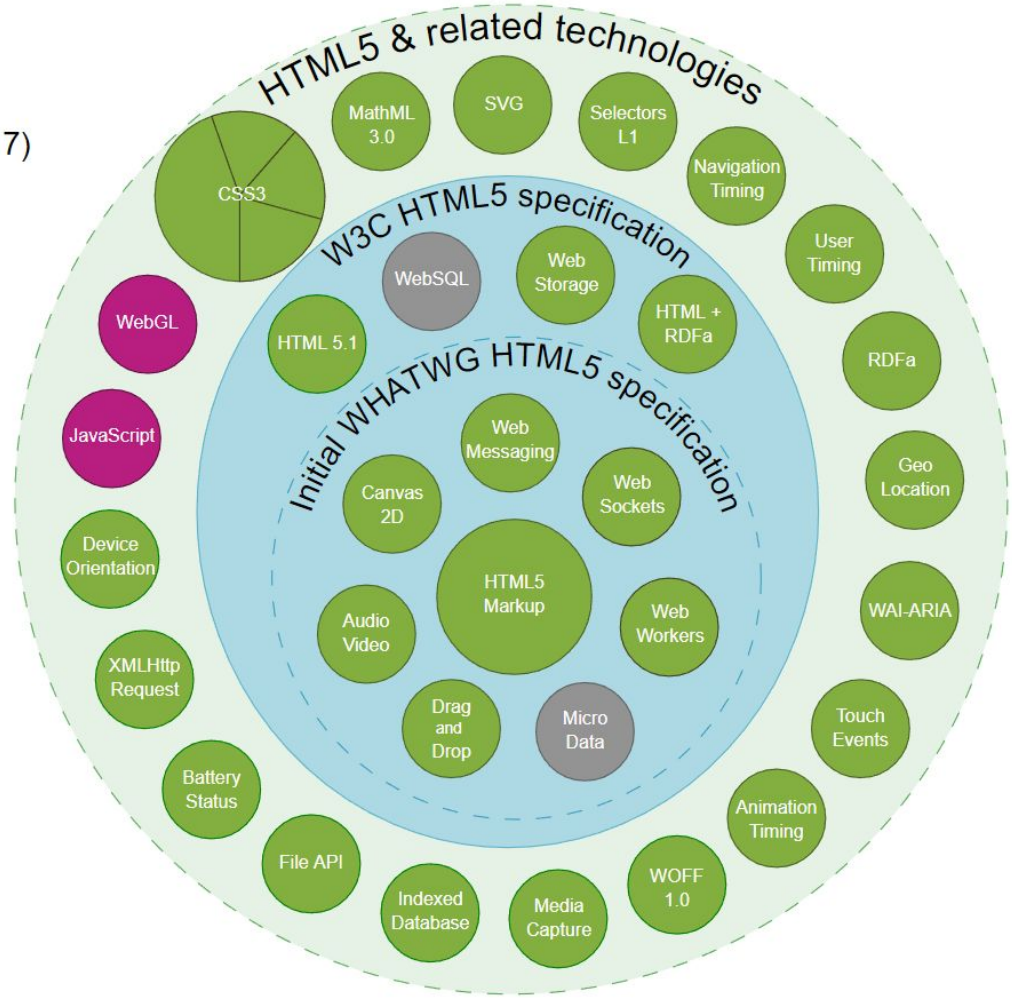
Name	Description	Argument type	Argument name
attachEvent	Similar to W3C's addEventListener method.	String	sEvent
		Pointer	fpNotify
detachEvent	Similar to W3C's removeEventListener method.	String	sEvent
		Pointer	fpNotify
fireEvent	Similar to W3C's dispatchEvent method.	String	sEvent
		Event	oEventObject

Mais
aujourd’hui !

HTML5.1

Taxonomy & Status (September 2017)

- Recommendation/Proposed
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Depreceted or inactive



⟶ ES6 ou EcmaScript 2015

⟶ *class, extends, super, getter/setter, static, Map, Set, Promise ...*

⟶ **Couverture** : 97%

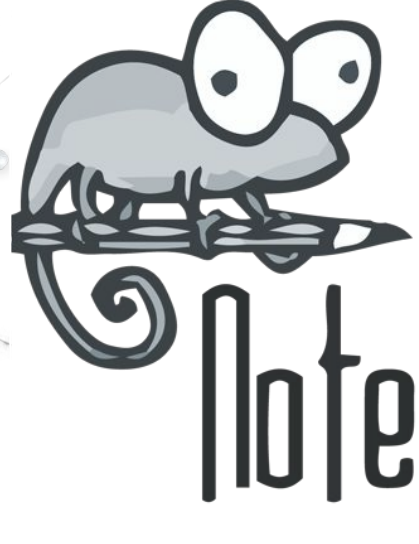
⟶ IE 11 ⟶ 2013

	Desktop browsers							
	11%	93%	96%	94%	97%	97%	99%	99%
Feature name	IE 11	Edge 14	Edge 15	FF 52 ESR	FF 55	CH 60, OP 47[1]	SF 10	SF 10.1
default function parameters	0/7	7/7	7/7	6/7	7/7	7/7	7/7	7/7
rest parameters	0/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5
spread (...) operator	0/15	15/15	15/15	15/15	15/15	15/15	15/15	15/15
object literal extensions	0/6	6/6	6/6	6/6	6/6	6/6	6/6	6/6
for..of loops	0/9	7/9	9/9	7/9	9/9	9/9	9/9	9/9
octal and binary literals	0/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4
template literals	0/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5
RegExp "y" and "u" flags	0/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5
destructuring, declarations	0/22	21/22	22/22	21/22	22/22	22/22	22/22	22/22
destructuring, assignment	0/24	23/24	24/24	23/24	24/24	24/24	24/24	24/24
destructuring, parameters	0/24	22/24	23/24	21/24	24/24	24/24	24/24	24/24
Unicode code point escapes	0/2	2/2	2/2	1/2	2/2	2/2	2/2	2/2
new.target	0/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2
Bindings								
const	12/16	16/16	16/16	16/16	16/16	16/16	16/16	16/16
let	10/12	12/12	12/12	12/12	12/12	12/12	12/12	12/12
block-level function declaration[11]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Functions								
arrow functions	0/13	13/13	13/13	13/13	13/13	13/13	13/13	13/13
class	0/24	24/24	24/24	24/24	24/24	24/24	24/24	24/24
super	0/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8
generators	0/27	27/27	27/27	25/27	27/27	27/27	27/27	27/27
Built-ins								
typed arrays	16/46	46/46	46/46	46/46	46/46	46/46	46/46	46/46
Map	8/19	18/19	19/19	18/19	19/19	19/19	19/19	19/19
Set	8/19	18/19	19/19	18/19	19/19	19/19	19/19	19/19

Source : http://kangax.github.io/compat-table/es6/

Osons

Javascript Vanilla !



→ **IDE** *(notepad++ ou autre)*

→ **Navigateur** *(chrome)*

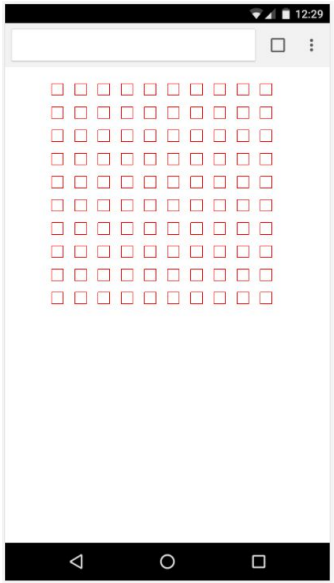
→ **Serveur web** *(wamp, xamp, ...)*

“ Par où commencer ? ”

Vous dans 10 minutes...

1

- Point & Polygon
- generatePolygons()



```html
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
        <meta content='user-scalable=0' name='viewport'/>
        <style>
            * {
                padding: 0;
                margin: 0;
            }
        </style>
    </head>
    <body>
        <canvas id="myCanvas" width="500" height="500"></canvas>
        <script>"use strict";
            /**
             *  Entry point
             */
            window.addEventListener("DOMContentLoaded", function() {
                const canvas = document.getElementById("myCanvas");
                const ctx = canvas.getContext("2d");
                ...
            });

            /**
             *  Javascript Model
             */
            class Polygon {
                ...
            }
        </script>
    </body>
</html>
```

NOT IMPRESSED

```html
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
        <meta content='user-scalable=0' name='viewport'/>
        <style>
            * {
                padding: 0;
                margin: 0;
            }
        </style>
    </head>
    <body>
        <canvas id="myCanvas" width="500" height="500"></canvas>
        <script>"use strict";
            /**
             *  Entry point
             */
            window.addEventListener("DOMContentLoaded", function() {
                const canvas = document.getElementById("myCanvas");
                const ctx = canvas.getContext("2d");
                ...
            });

            /**
             *  Javascript Model
             */
            class Polygon {
                ...
            }
        </script>
    </body>
</html>
```

CSS : File or Inline ?

→ Inline si poids page **< 13.8 Ko**

→ Inline si **style critique**

→ Sinon file **async** *(link rel dans body)*

NOT IMPRESSED

```html
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
        <meta content='user-scalable=0' name='viewport'/>
        <style>
            * {
                padding: 0;
                margin: 0;
            }
        </style>
    </head>
    <body>
        <canvas id="myCanvas" width="500" height="500"></canvas>
        <script>"use strict";
            /**
             *  Entry point
             */
            window.addEventListener("DOMContentLoaded", function() {
                const canvas = document.getElementById("myCanvas");
                const ctx = canvas.getContext("2d");
                ...
            });

            /**
             *  Javascript Model
             */
            class Polygon {
                ...
            }
        </script>
    </body>
</html>
```

```js
"use strict";
varialeMalDéclarée = 17; // lève une ReferenceError
```

JS :

→ Rend **explicite erreur** normalement silencieuse

→ Exécution **plus rapide** du js

→ **Interdit** Next ES keyword

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Strict_mode

NOT IMPRESSED

```html
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
        <meta content='user-scalable=0' name='viewport'/>
        <style>
            * {
                padding: 0;
                margin: 0;
            }
        </style>
    </head>
    <body>
        <canvas id="myCanvas" width="500" height="500"></canvas>
        <script>"use strict";
            /**
             *  Entry point
             */
            window.addEventListener("DOMContentLoaded", function() {
                const canvas = document.getElementById("myCanvas");
                const ctx = canvas.getContext("2d");
                ...
            });

            /**
             *  Javascript Model
             */
            class Polygon {
                ...
            }
        </script>
    </body>
</html>
```

JS :

→ Equivalent “***main***” java

→ **Fin** du <body> !

→ **“DomContentLoaded”** et pas ***“load”*** !

NOT IMPRESSED

DOMContentLoaded

Cette traduction est incomplète. Aidez à traduire cet article depuis l'anglais.

L'événement `DOMContentLoaded` est déclenché lorsque le document HTML initial a été complètement chargé et analysé, sans attendre les feuilles de style, images et sous-trames pour terminer le chargement. Un événement très différente - `load` - doit être utilisé seulement pour détecter une page entièrement chargée. C'est une erreur très répandue d'utiliser `load` quand `DOMContentLoaded` serait beaucoup plus approprié, alors soyez prudents.

```
<script>
"use strict";
    /**
     *  Entry point, initialize PolygonManager and listeners
     */
    window.addEventListener("DOMContentLoaded", function () {
        // Get canvas and draw polygons
        const canvas = document.getElementById("fisheye");
        const ctx = canvas.getContext("2d");
        ...
    });
<script>
```

```html
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
        <meta content='user-scalable=0' name='viewport'/>
        <style>
            * {
                padding: 0;
                margin: 0;
            }
        </style>
    </head>
    <body>
        <canvas id="myCanvas" width="500" height="500"></canvas>
        <script>
            /**
             *  Entry point
             */
            window.addEventListener("DOMContentLoaded", function() {
                const canvas = document.getElementById("myCanvas");
                const ctx = canvas.getContext("2d");
                ...
            });

            /**
             *  Javascript Model
             */
            class Polygon {
                ...
            }
        </script>
    </body>
</html>
```

JS : File or Inline ?

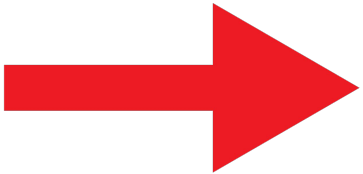
→ Inline si poids page **< 13.8 Ko**

→ Inline si **js critique**

→ Sinon file

Javascript ES6

```
/**
 *  Point
 *  @class
 */
class Point {
    /**
     * @param {Number} x
     * @param {Number} y
     */
    constructor(x, y) {
        this.x = x;
        this.y = y;
    }
    /**
     * @param {Number} a
     * @param {Number} b
     */
    static distance(a, b) {
        const dx = a.x - b.x;
        const dy = a.y - b.y;
        return Math.hypot(dx, dy);
    }
}
```



```
let p1 = new Point(10,10),
    p2 = new Point(20,20);

p1.x = p2.x;

const dist = Point.distance(p1,p2);
```

```html
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Hello World</title>
        <meta content='user-scalable=0' name='viewport'/>
        <style>
            * {
                padding: 0;
                margin: 0;
            }
        </style>
    </head>
    <body>
        <canvas id="myCanvas" width="500" height="500"></canvas>
        <script>
            /**
             *  Entry point
             */
            window.addEventListener("DOMContentLoaded", function() {
                const canvas = document.getElementById("myCanvas");
                const ctx = canvas.getContext("2d");
                ...
            });

            /**
             *  Javascript Model
             */
            class Polygon {
                ...
            }
        </script>
    </body>
</html>
```

Canvas : zone de dessin

HTML

```html
<canvas id="canvas" width="100" height="100"></canvas>

<input id="r" type="range" value="250" min="40" max="500">

<canvas id="canvas2" width="300" height="300"></canvas>
```

```
input {display:block}
```

JAVASCRIPT

```javascript
const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");

ctx.rect(35,35,50,50); // (x,y,width,height)
ctx.stroke();

// draw line
ctx.beginPath();
ctx.moveTo(35,35); // start to (x,y)
ctx.lineTo(85,85); // go to (x,y)
ctx.lineWidth = 2;
ctx.strokeStyle = "red";
ctx.stroke();

const canvas2 = document.getElementById("canvas2");
const ctx2 = canvas2.getContext("2d");

// load image in canvas
let img = new Image();
img.src = "https://www.gstatic.com/webp/gallery3/1_webp_a.png";
img.onload = function () {
  ctx2.drawImage(img, 0, 0);
};
// Get pixels from canvas
let data = ctx2.getImageData(0,0,canvas2.width,canvas2.height); // (x,y,width,height)
// Replace them
ctx2.putImageData(data, 0, 0); // (ImageData,x,y)
```

HTML

```html
<canvas id="canvas" width="100" height="100"></canvas>

<input id="r" type="range" value="250" min="40" max="500">

<canvas id="canvas2" width="300" height="300"></canvas>
```

```
input {display:block}
```

JAVASCRIPT

```javascript
const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");

ctx.rect(35,35,50,50); // (x,y,width,height)
ctx.stroke();

// draw line
ctx.beginPath();
ctx.moveTo(35,35); // start to (x,y)
ctx.lineTo(85,85); // go to (x,y)
ctx.lineWidth = 2;
ctx.strokeStyle = "red";
ctx.stroke();

const canvas2 = document.getElementById("canvas2");
const ctx2 = canvas2.getContext("2d");

// load image in canvas
let img = new Image();
img.src = "https://www.gstatic.com/webp/gallery3/1_webp_a.png";
img.onload = function () {
  ctx2.drawImage(img, 0, 0);
};
// Get pixels from canvas
let data = ctx2.getImageData(0,0,canvas2.width,canvas2.height); // (x,y,width,height)
// Replace them
ctx2.putImageData(data, 0, 0); // (ImageData,x,y)
```

HTML

```html
<canvas id="canvas" width="100" height="100"></canvas>

<input id="r" type="range" value="250" min="40" max="500">

<canvas id="canvas2" width="300" height="300"></canvas>
```

```
input {display:block}
```

JAVASCRIPT

```javascript
const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");

ctx.rect(35,35,50,50); // (x,y,width,height)
ctx.stroke();

// draw line
ctx.beginPath();
ctx.moveTo(35,35); // start to (x,y)
ctx.lineTo(85,85); // go to (x,y)
ctx.lineWidth = 2;
ctx.strokeStyle = "red";
ctx.stroke();

const canvas2 = document.getElementById("canvas2");
const ctx2 = canvas2.getContext("2d");

// load image in canvas
let img = new Image();
img.src = "https://www.gstatic.com/webp/gallery3/1_webp_a.png";
img.onload = function () {
  ctx2.drawImage(img, 0, 0);
};
// Get pixels from canvas
let data = ctx2.getImageData(0,0,canvas2.width,canvas2.height); // (x,y,width,height)
// Replace them
ctx2.putImageData(data, 0, 0); // (ImageData,x,y)
```

HTML

```html
<canvas id="canvas" width="100" height="100"></canvas>

<input id="r" type="range" value="250" min="40" max="500">

<canvas id="canvas2" width="300" height="300"></canvas>
```

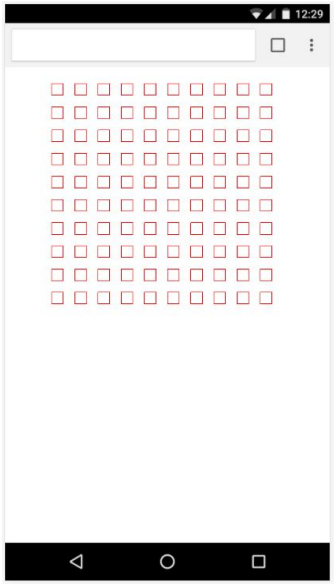
```
input {display:block}
```

JAVASCRIPT

```javascript
const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");

ctx.rect(35,35,50,50); // (x,y,width,height)
ctx.stroke();

// draw line
ctx.beginPath();
ctx.moveTo(35,35); // start to (x,y)
ctx.lineTo(85,85); // go to (x,y)
ctx.lineWidth = 2;
ctx.strokeStyle = "red";
ctx.stroke();

const canvas2 = document.getElementById("canvas2");
const ctx2 = canvas2.getContext("2d");

// load image in canvas
let img = new Image();
img.src = "https://www.gstatic.com/webp/gallery3/1_webp_a.png";
img.onload = function () {
  ctx2.drawImage(img, 0, 0);
};
// Get pixels from canvas
let data = ctx2.getImageData(0,0,canvas2.width,canvas2.height); // (x,y,width,height)
// Replace them
ctx2.putImageData(data, 0, 0); // (ImageData,x,y)
```

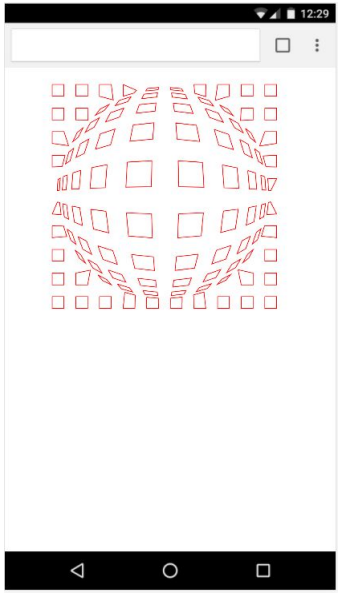
1

- Point & Polygon
- generatePolygons()



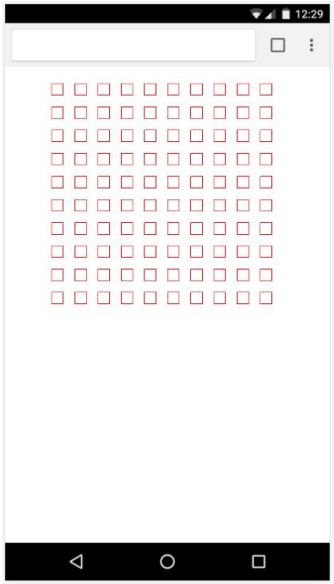
2

Fisheye deformation



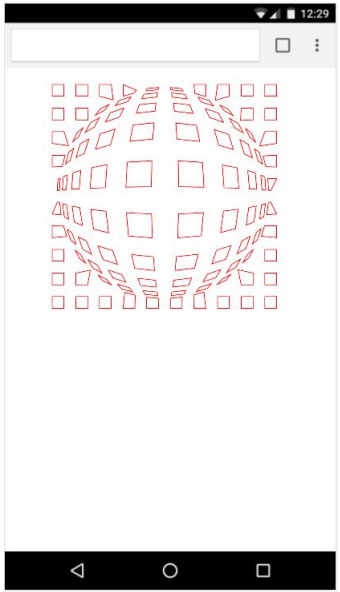
1

- Point & Polygon
- generatePolygons()



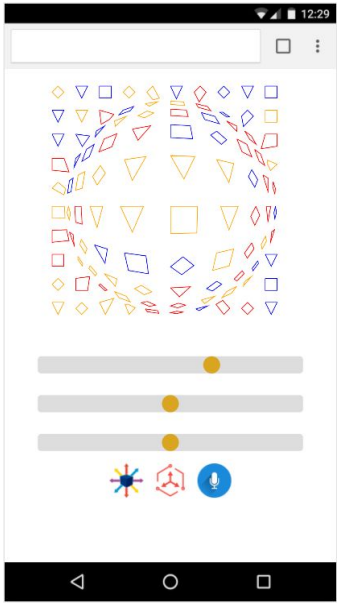
2

Fisheye deformation



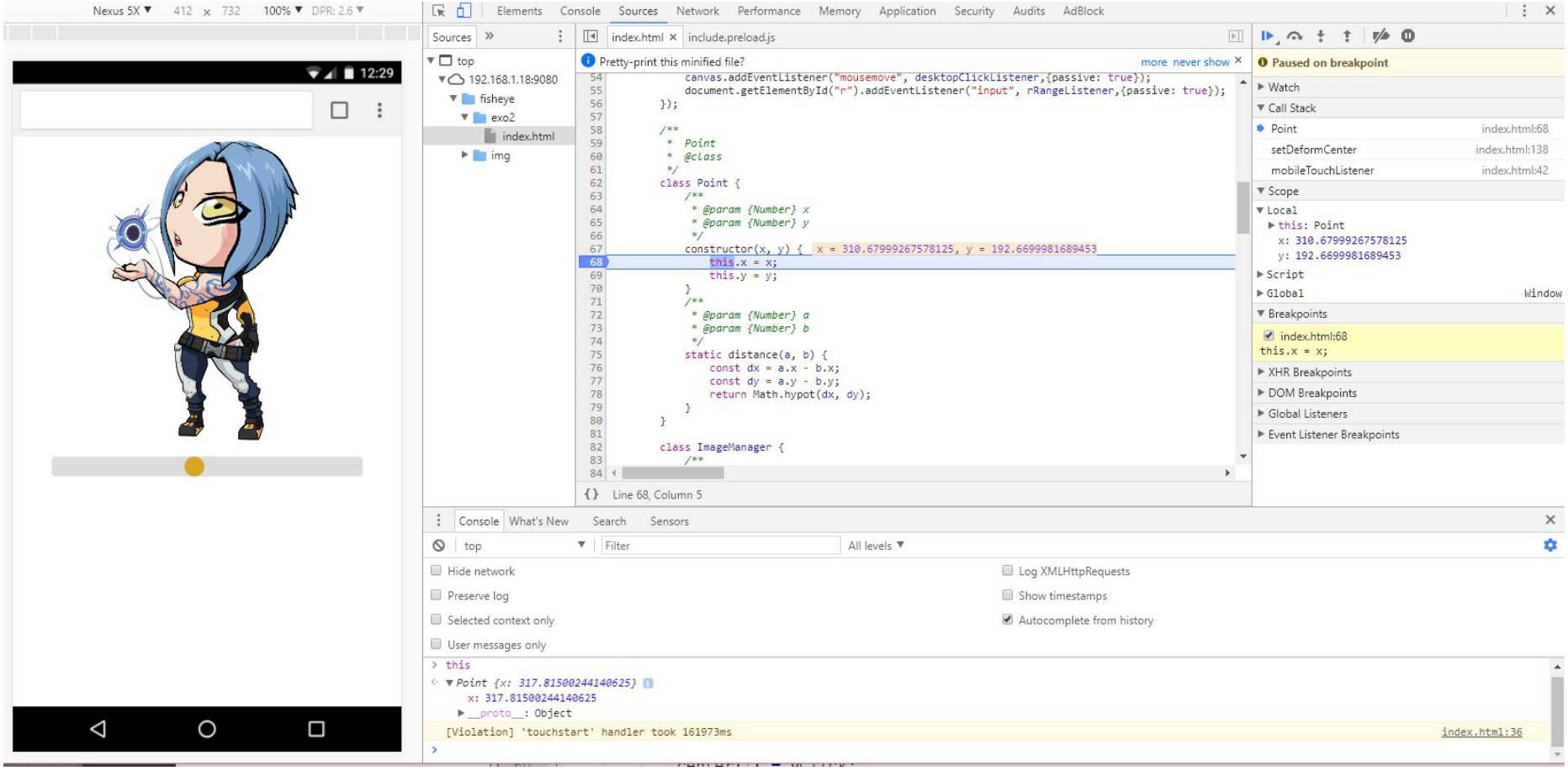
3

Listeners

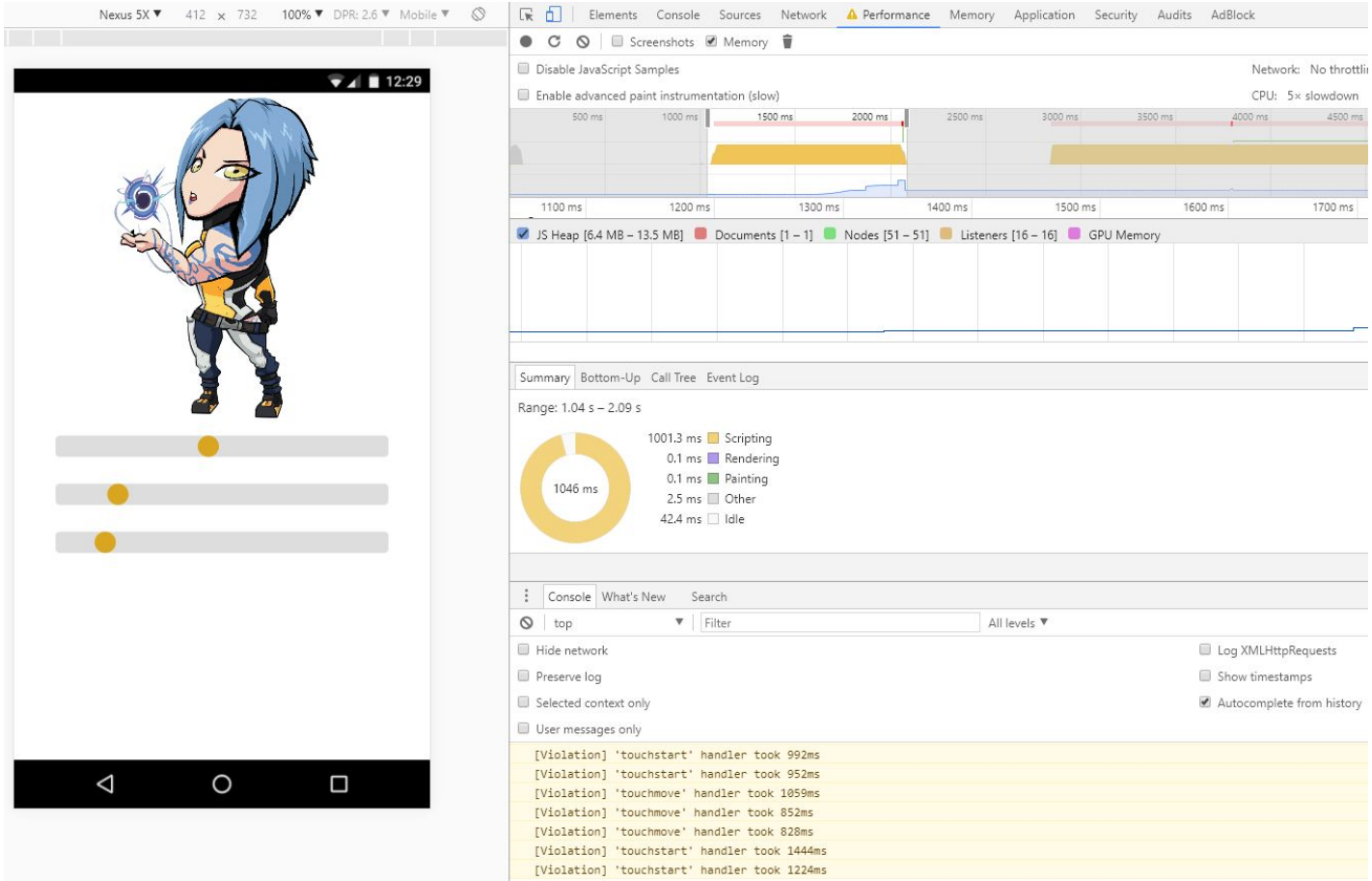


One more thing...

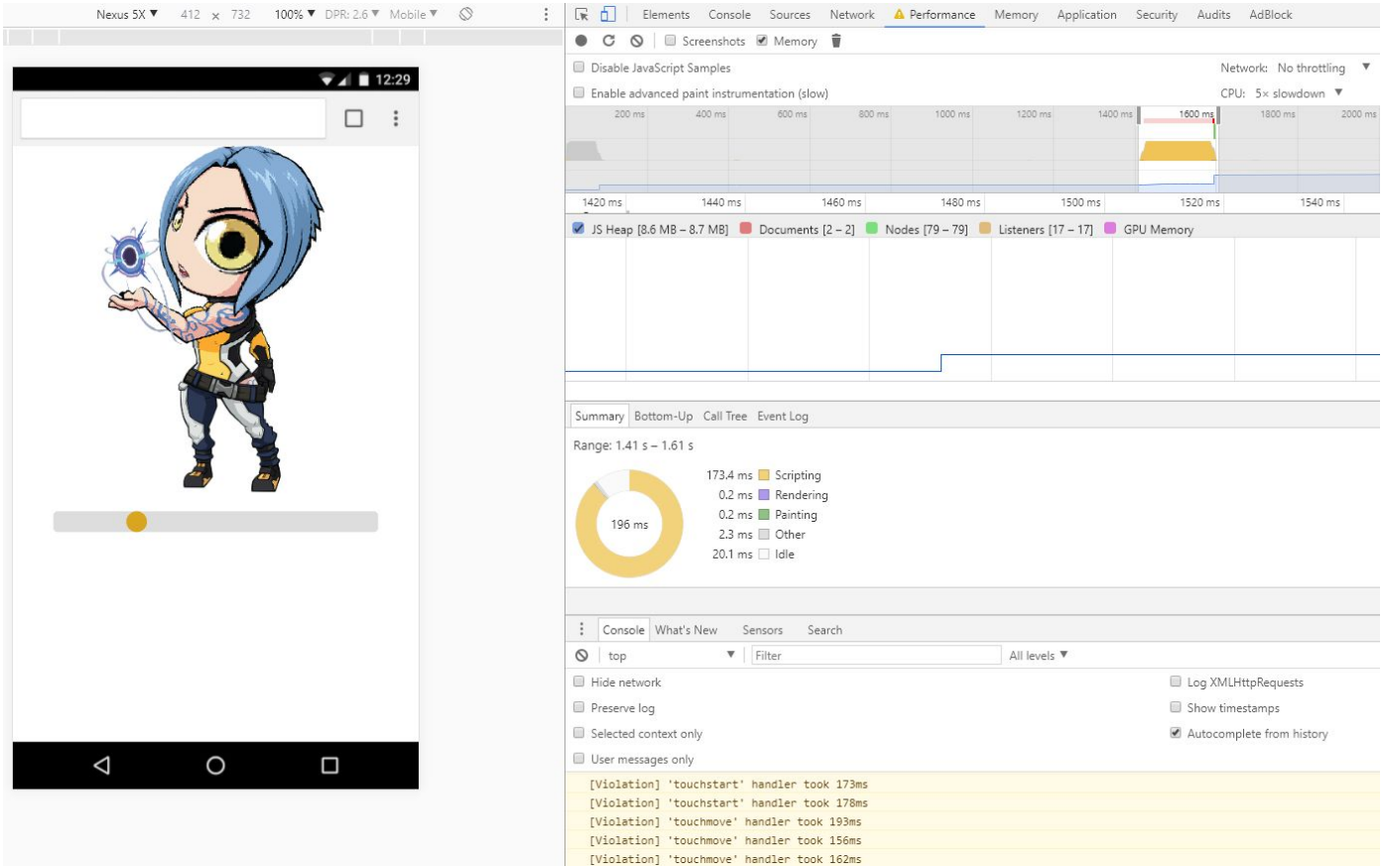
Devtools : Debug



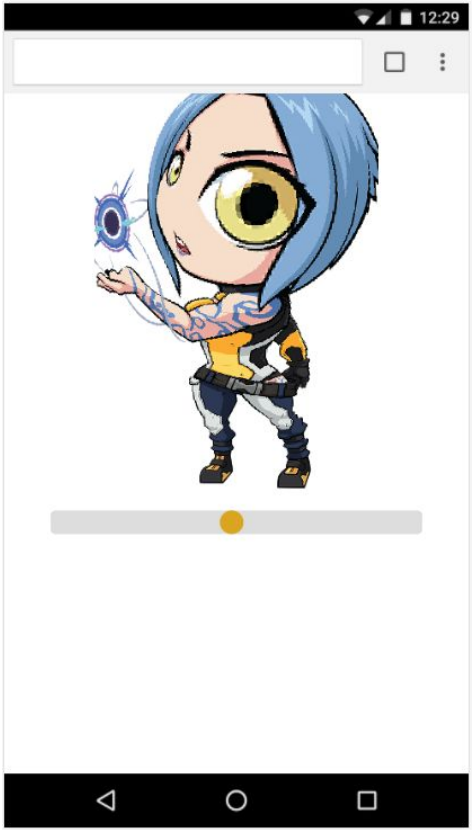
Devtools : Performance



Devtools : Performance



Fisheye image (performance)



→ Utilisation de *“shaders”* mais pour le TD on reste en canvas ‘2d’

```
/**
 * Deform Image with Fisheye formula
 */
fishEyeDeform () {
    const data = this.img.data;
    const width = this.canvas.width;
    const R = ImageManager.r;
    const deformCenterX = this.deformCenter.x;
    const deformCenterY = this.deformCenter.y;
    // Iterate on pixels array
    let i = data.length;
    while (i-=4) {
        const x = (i / 4) % width;
        const y = Math.floor((i / 4) / width);
        const dx = deformCenterX - x;
        const dy = deformCenterY - y;
        const dist = Math.hypot(dx, dy);
        // More perf
        if (dist < R) {
            // Compute new X and Y
            const scale = Math.sin(dist/R*Math.PI/2);
            const x2 = Math.round(deformCenterX - dx * scale);
            const y2 = Math.round(deformCenterY - dy * scale);
            const i2 = (y2 * width + x2) * 4;
            this.deformImg.data[i  ] = data[i2];
            this.deformImg.data[i+1] = data[i2+1];
            this.deformImg.data[i+2] = data[i2+2];
            this.deformImg.data[i+3] = data[i2+3];
        }
    }
}
```

Règle n°1 :
Optimisation ssi nécessaire !

Règle n°2 :
Ne sacrifiez pas la lisibilité du code !

Règle n°3 :
Factoriser l'algorithme avant le langage !

Optimisations faites :

pixels / 2	=> 30%
formule	=> 30%
inline	=> 20%
while	=> 10%