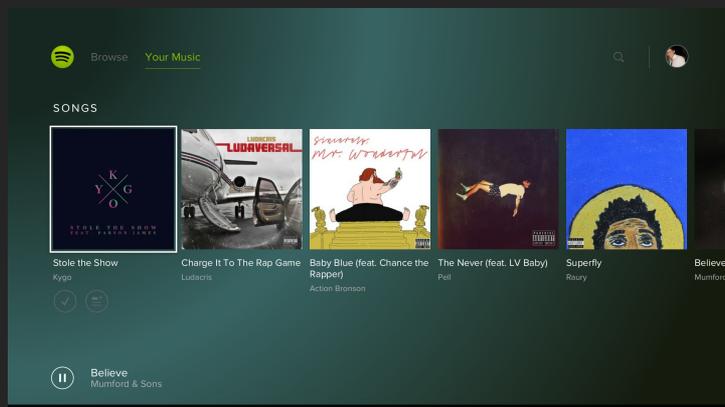
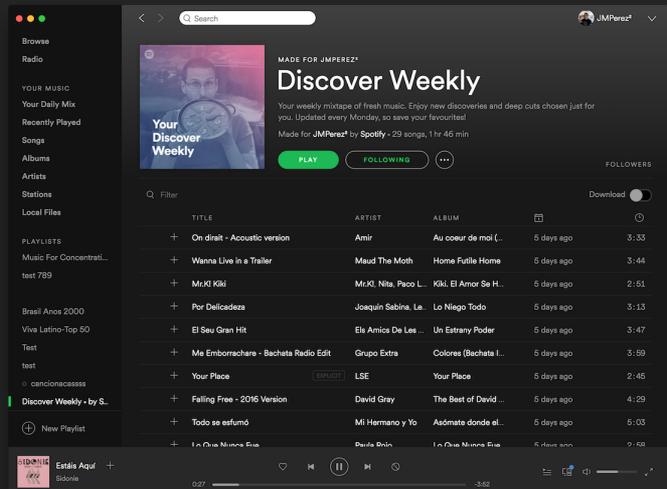
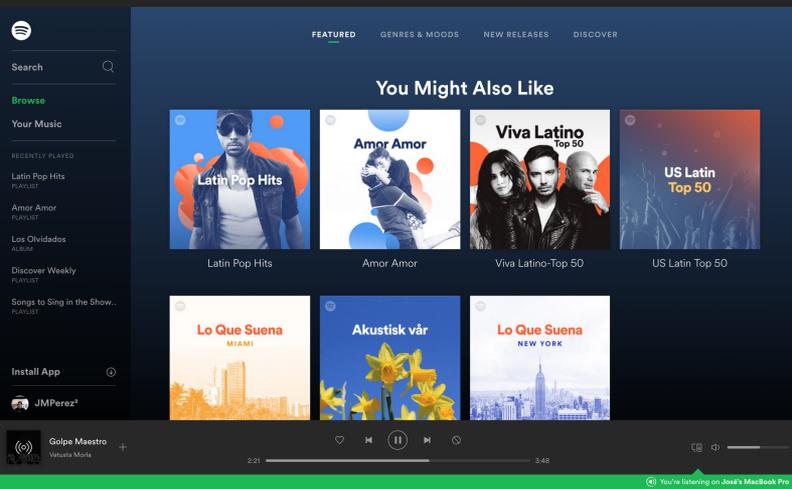


# Cargando imágenes de una forma óptima

José M. Pérez · Desarrollador Web en Spotify  
@jmperezperez

<http://spoti.fi/alicante-frontend>

# Qué hago





# Page Load Time



# Versión A

0.0s	0.3s	0.6s	0.9s	1.2s

# Versión B

0.0s	0.3s	0.6s	0.9s	1.2s

# Rendimiento percibido por el usuario (User Perceived Performance)

$$SpeedIndex = \int_0^{fin} 1 - \frac{R}{100}$$

fin = tiempo final en milisegundos

R = % de pantalla renderizado

# WebPageTest

Tested From: Dulles, VA - iPhone 6 iOS 9 - 3GFast

1: reactalicante.  
es/  
(Edit)





Developer Tools - http://reactalicante.es/

Elements Console Sources Network Performance Memory Application >> 7 1

View: Group by frame Preserve log Disable cache Offline No throttling

31 ms 64 ms 118 ms 163 ms 852 ms 1.34 s 2.14 s 2.28 s 2.55 s

1000 ms 2000 ms 3000 ms 4000 ms 5000 ms 6000 ms 7000 ms 8000 ms 9000 ms

Name

- map.js
- util.js
- marker
- csi?v=
- balsam
- brains-
- flywire.
- lemonc
- nitsnet
- nativeb
- algolia.
- alicante
- jetbrair
- oreilly.g
- sticker
- survive
- 33b95c
- 4638b2

2.00 s

2.14 s

a65302dfa79acfbe6cc8...	200	jpeg	(index)	105 KB	2...	
c8ce70fb5975333076d2...	200	jpeg	(index)	75.4 KB	2...	
close.png	200	png	(index)	652 B	1...	
loading.gif	200	gif	(index)	8.6 KB	1...	
prev.png	200	png	(index)	1.7 KB	1...	
next.png	200	png	(index)	1.7 KB	1...	

83 / 113 requests | 2.3 MB / 2.6 MB transferred | Finish: 8.66 s | DOMContentLoaded: 1.91 s | Load: 3.61 s

# Más kilobytes

*puede, a veces,*

# mejorar el rendimiento percibido

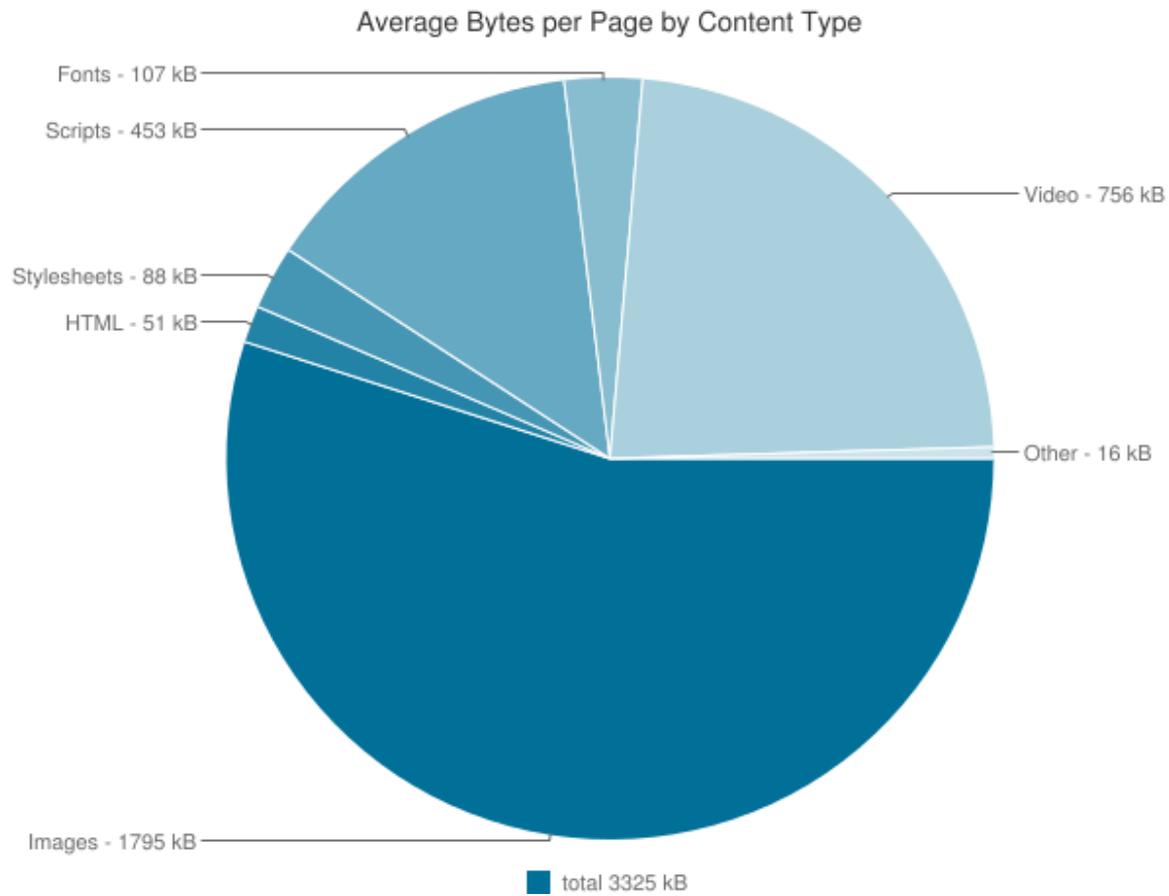
# Evitando esto:



# Técnicas para mejorar el rendimiento percibido

- Server-side rendering
- Critical CSS
- JS Async
- Fuentes Async

# ¿Qué pasa con las imágenes?



**imágenes**  
**1795kB (54%)**

**vídeos**  
**756kB (23%)**

**scripts**  
**453kB (14%)**

fuentes: <http://archive> 15 agosto 2017





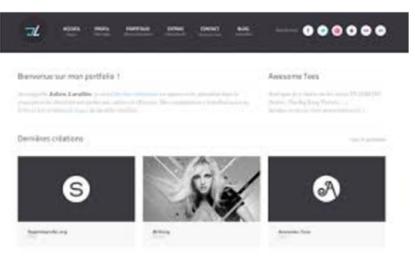
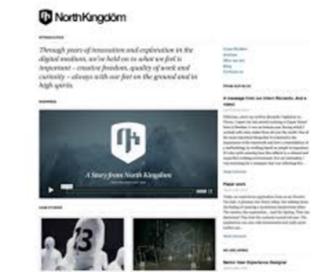
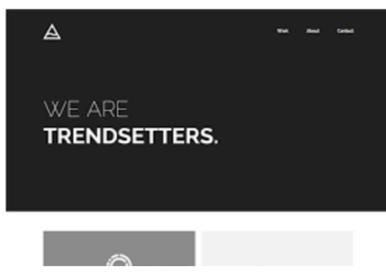
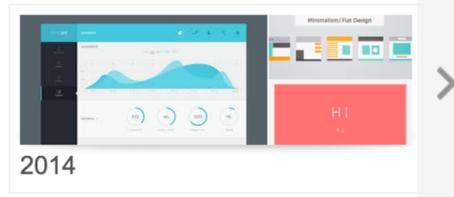
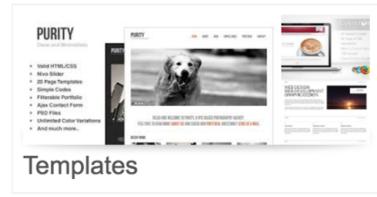
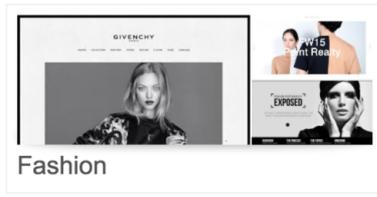
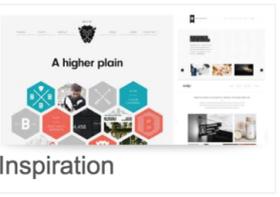
Usar o no usar  
imágenes

**CONSEJO 1**

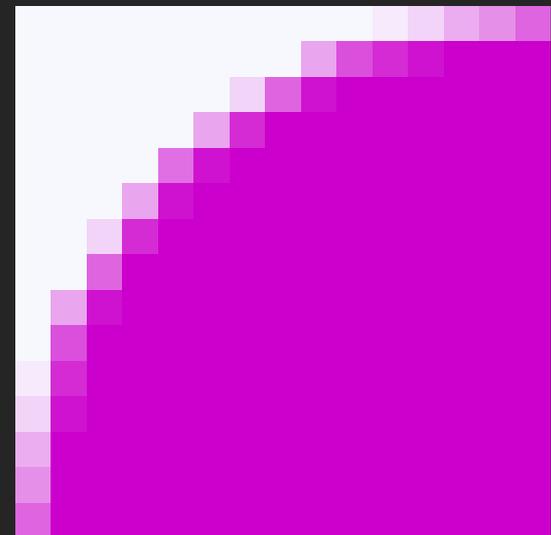
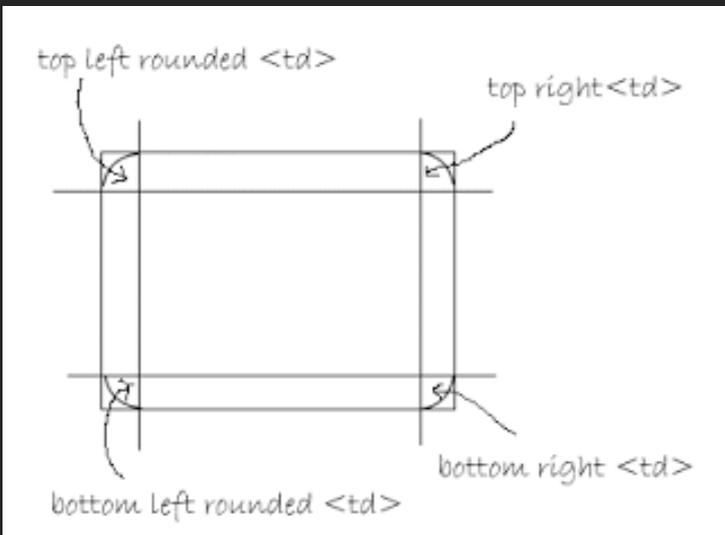
# Reduce las peticiones de imágenes

en especial las de mapa de bits (JPG, PNG, WebP...)

# Minimalismo | Diseño Flat



# Hace no mucho tiempo...





# Hello

By Adele

2015 • 1 SONG, 4:55

PLAY ON SPOTIFY

1. Hello

4:55



Featured on  
25

## More by Adele





# Hello

By Adele

2015 • 1 SONG, 4:55

PLAY ON SPOTIFY

1. Hello

4:55

# Cómo NO ocultar imágenes (1/2)

## CSS

```
@media (max-width: 480px) {  
  img {  
    display: none;  
  }  
}
```

## HTML

```
 <!-- el navegador hará la petición -->
```

# Cómo NO ocultar imágenes (2/2)

## CSS

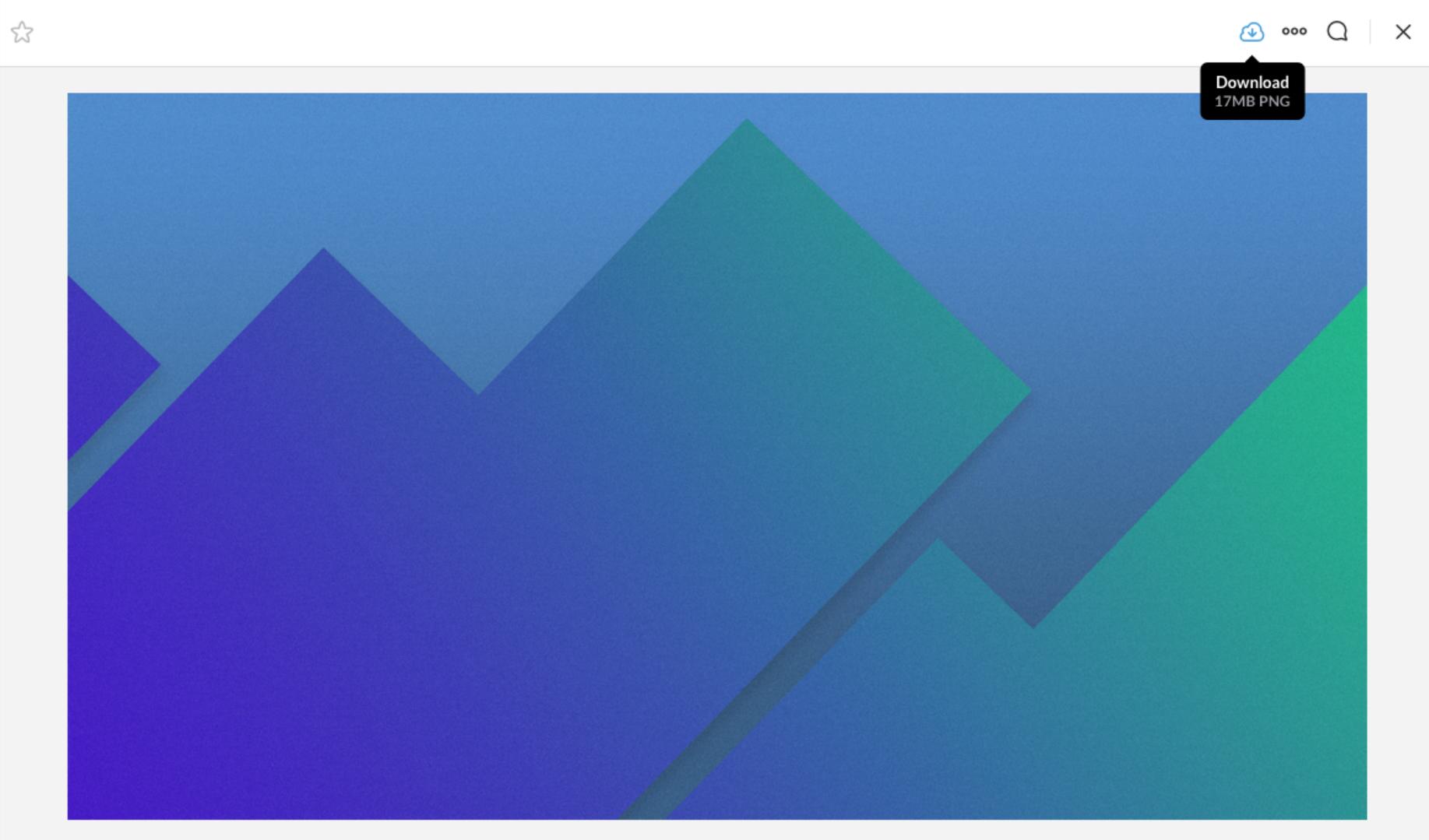
```
.hidden { display: none; }
```

## HTML

```
<body>  
  <section class="section-1">  
    <p>Esta es la sección actual</p>  
      
  </section>  
  <section class="section-2 hidden">  
    <p>Esta sección quizás se muestre luego</p>  
     <!-- el navegador pedirá esta imagen -->  
  </section>  
</body>
```

# CONSEJO 2

# Optimiza tus imágenes



Download  
17MB PNG

# formato adecuado

gif | png | jpg | webp | <lo\_que\_venga\_después>

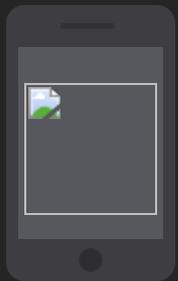
```
<picture>
  <source type="image/webp" srcset="2700x1209/mi-imagen.webp 2700w,
    1024x1024/mi-imagen.webp 1024w,
    600x600/mi-imagen.webp 600w"
    sizes="100vw" />
  <source srcset="2700x1209/mi-imagen.jpg 2700w,
    1024x1024/mi-imagen.jpg 1024w,
    600x600/mi-imagen.jpg 600w"
    sizes="100vw" />
  
</picture>
```

# compresión

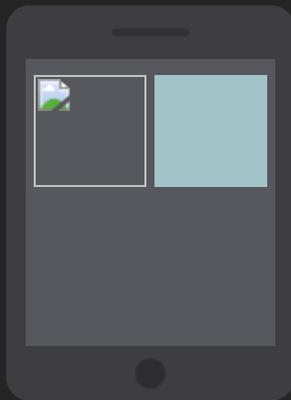
- sin pérdida (lossless)
  - perceptual

**CONSEJO 3**

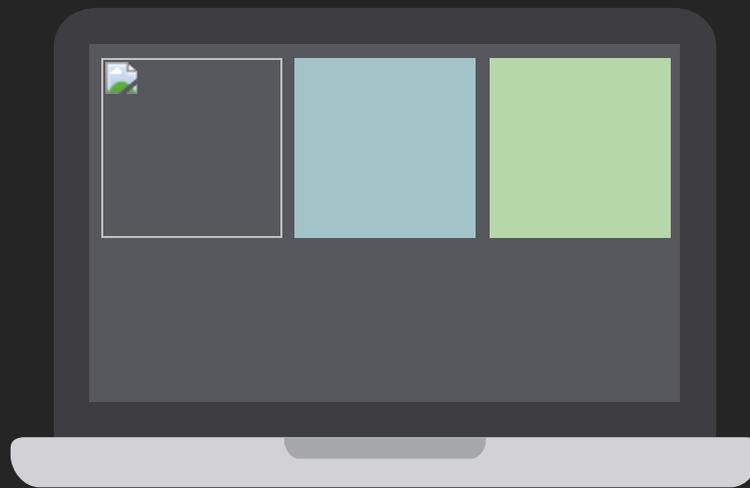
# Responsive



1 columna



2 columnas



3 columnas

IMÁGENES RESPONSIVE

# Ejemplo

```

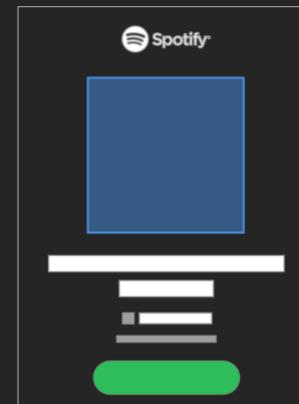
```

Reto: Mantener sincronizado el markup y el CSS

# CONSEJO 4

# Usar lazy-loading en imágenes

`<img>` encima del  
*fold*  
&  
*Lazy* debajo del *fold*

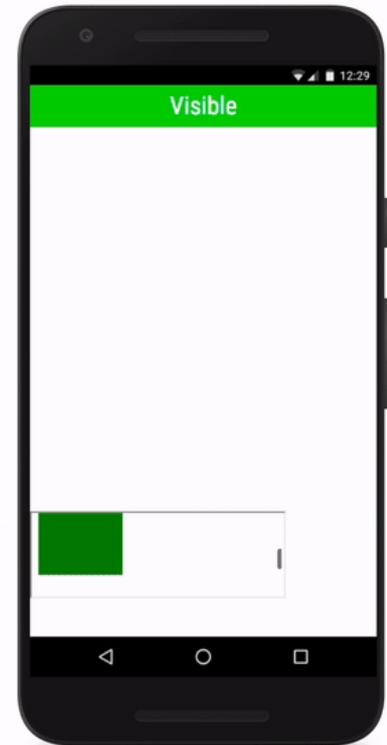


Above the  
fold content



Additional content  
by same artist

# Intersection Observer



INTERSECTION OBSERVER

# Ejemplo

```
// cargar imagen cuando esté a <= 100 px del viewport

const options = {
  rootMargin: '100px'
}

const callback = entries => {
  entries.forEach(entry => {
    if (entry.intersectionRatio > 0) {
      // cargar imagen
    }
  });
};

const observer = new IntersectionObserver(callback, options);
observer.observe(document.querySelector('.lazy-img'));
```

INTERSECTION OBSERVER

# Encapsulación en React

```
class LazyImage extends React.Component {

  constructor() {
    this.observer = new IntersectionObserver(entries => {
      if (entries[0].intersectionRatio > 0) {
        // ¡hacer la petición!
      }
    });
    this.element = null; /* render() le dará valor a través de un ref */
  }

  componentDidMount() {
    this.observer.observe(this.element);
  }

  componentWillUnmount() {
    this.observer.unobserve(this.element);
  }
  ...
}
```

INTERSECTION OBSERVER

# Soporte



# Qué mostrar mientras la imagen se está cargando

PLACEHOLDERS

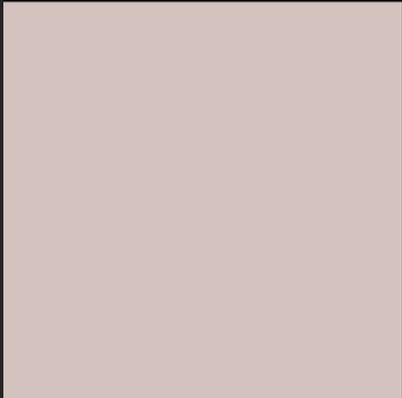
# Opciones



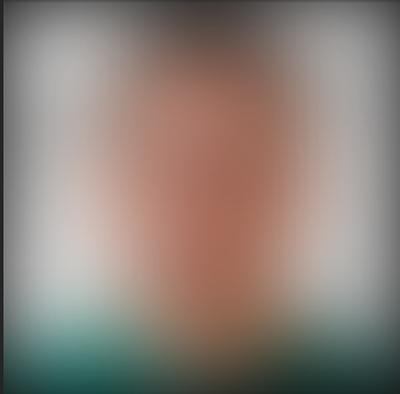
Nada



Placeholder

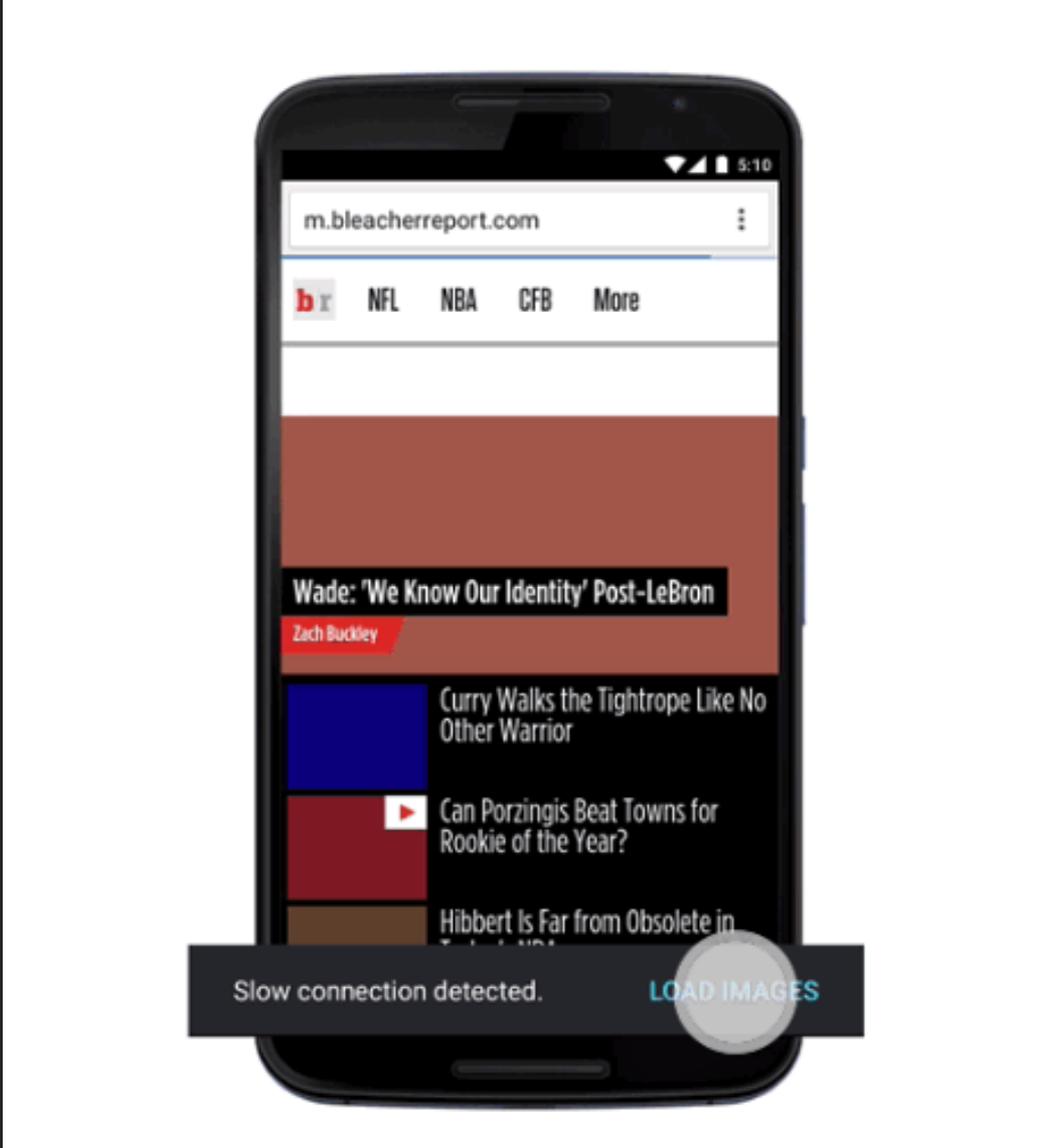


Color sólido



Carga progresiva o "Blur-up"

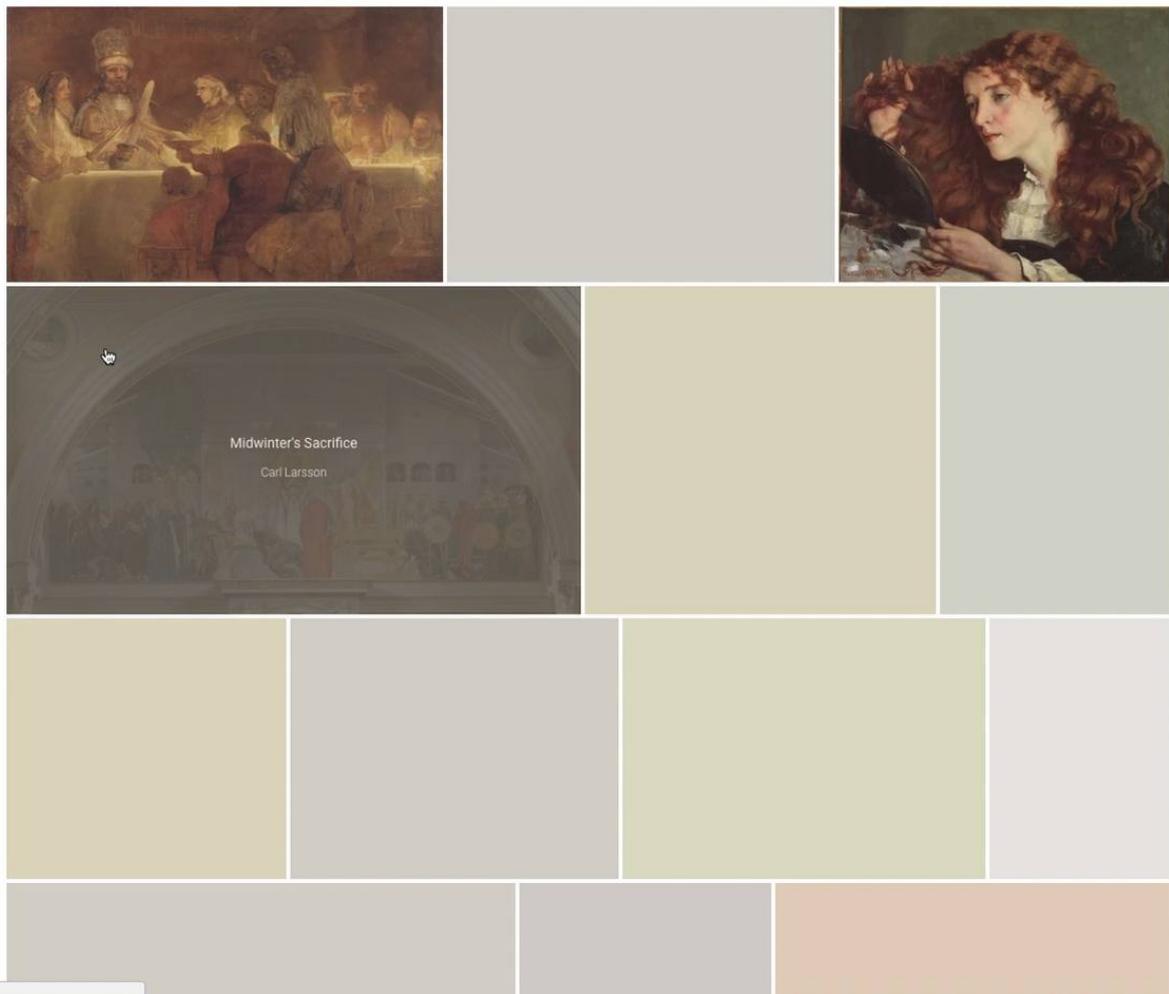
# Ejemplos de colores sólidos

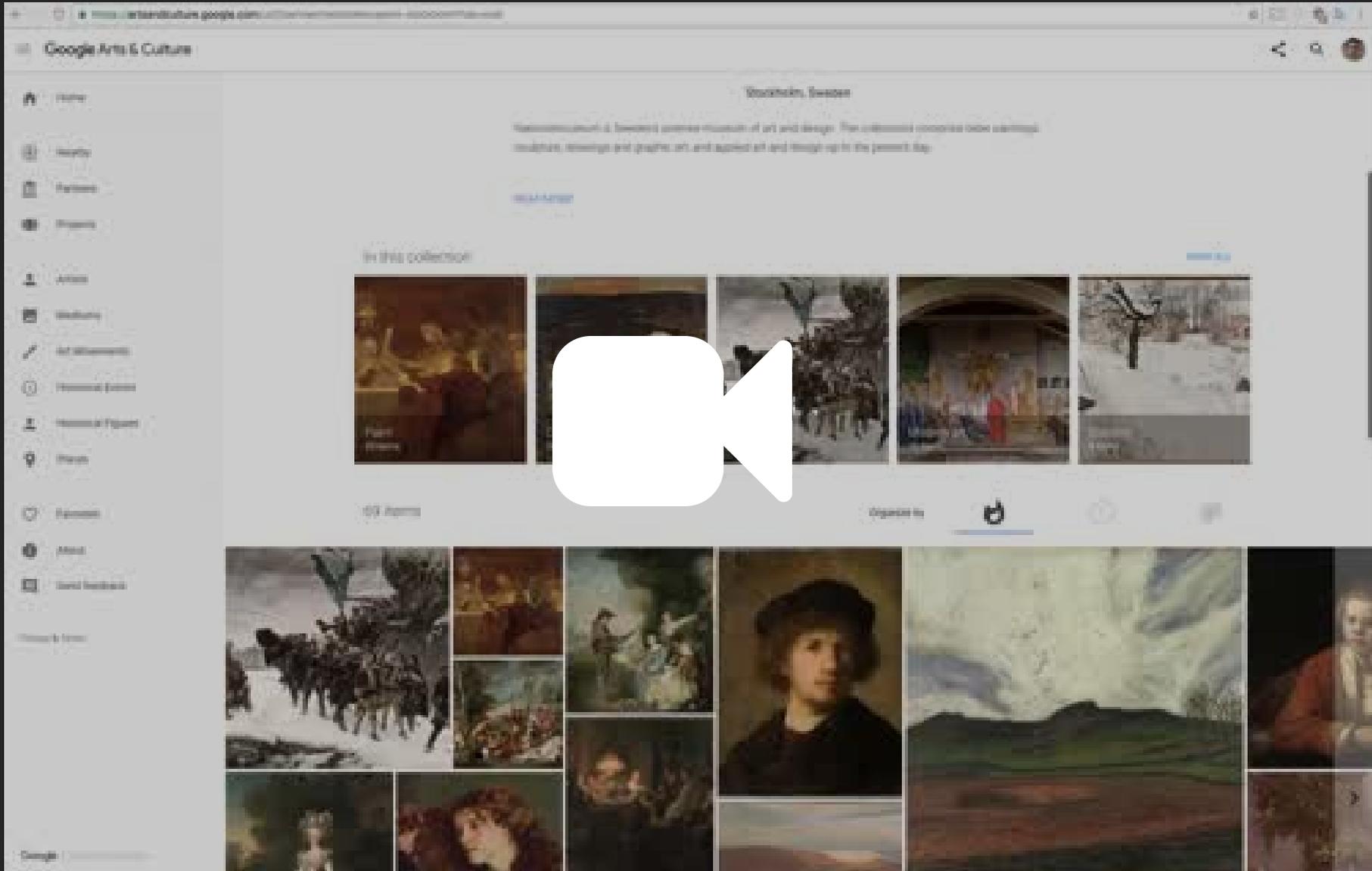


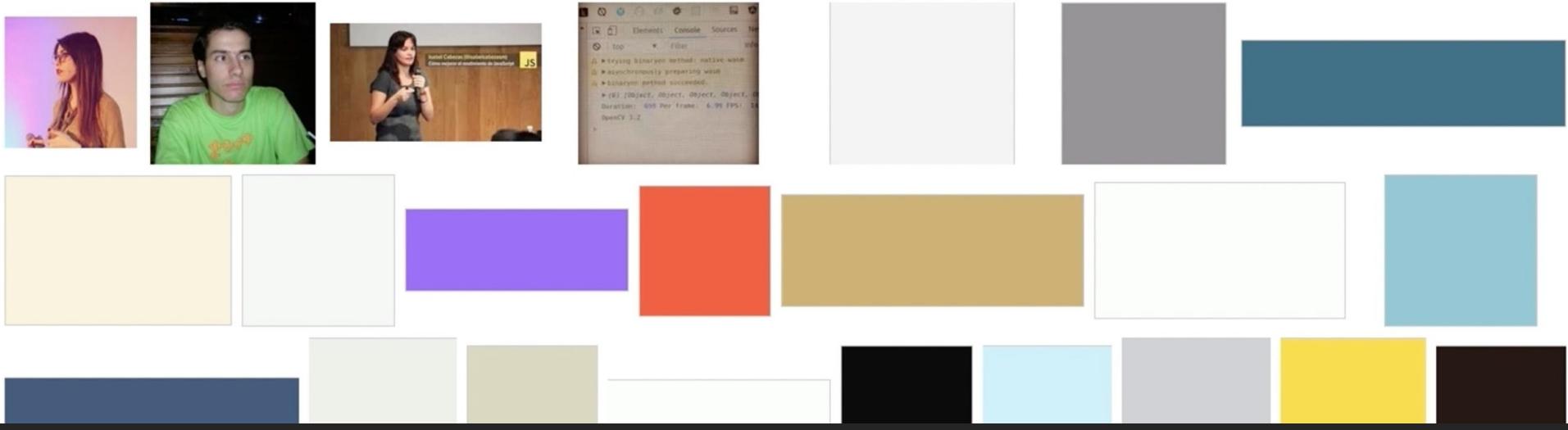
← Search Nationalmuseum Sweden | Paint

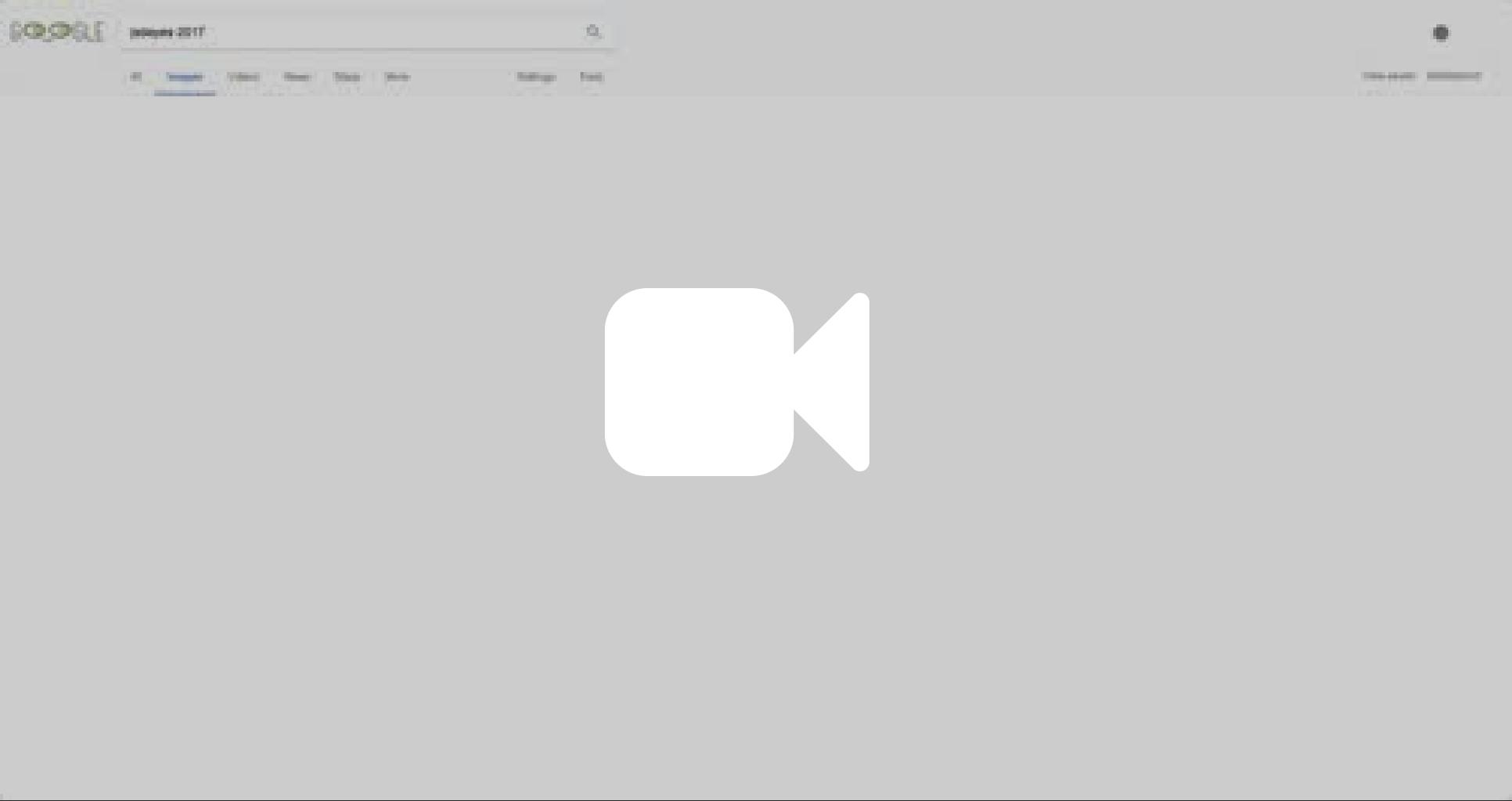
- 🏠 Home
- 📍 Nearby
- 🏛️ Partners
- 📅 Projects
- 👤 Artists
- 🖼️ Mediums
- 🖌️ Art Movements
- 🕒 Historical Events
- 👤 Historical Figures
- 📍 Places
- ❤️ Favorites
- 📄 About
- ☰ Send feedback
- Privacy & Terms

Nationalmuseum Sweden | Paint | 63 items



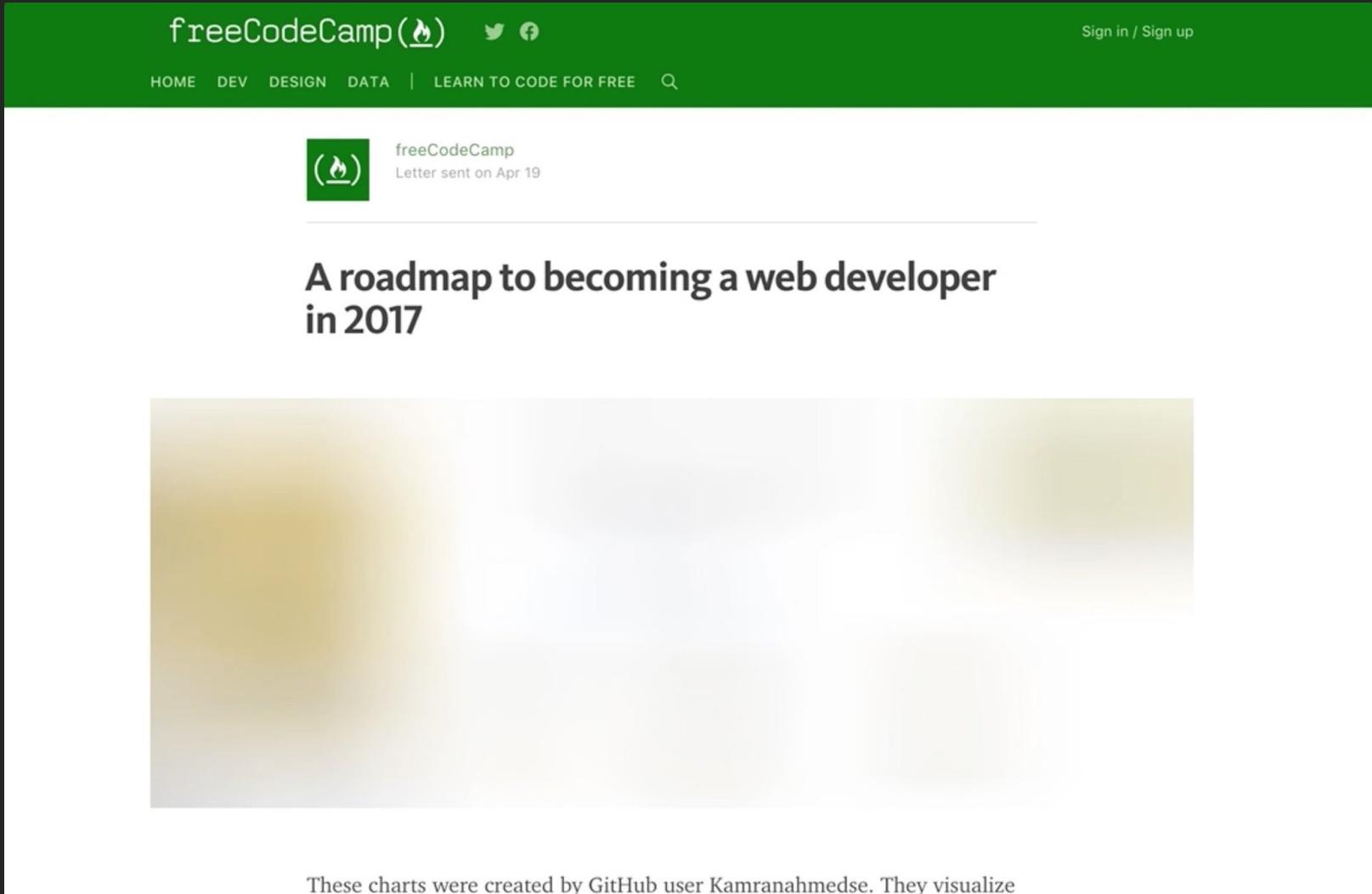






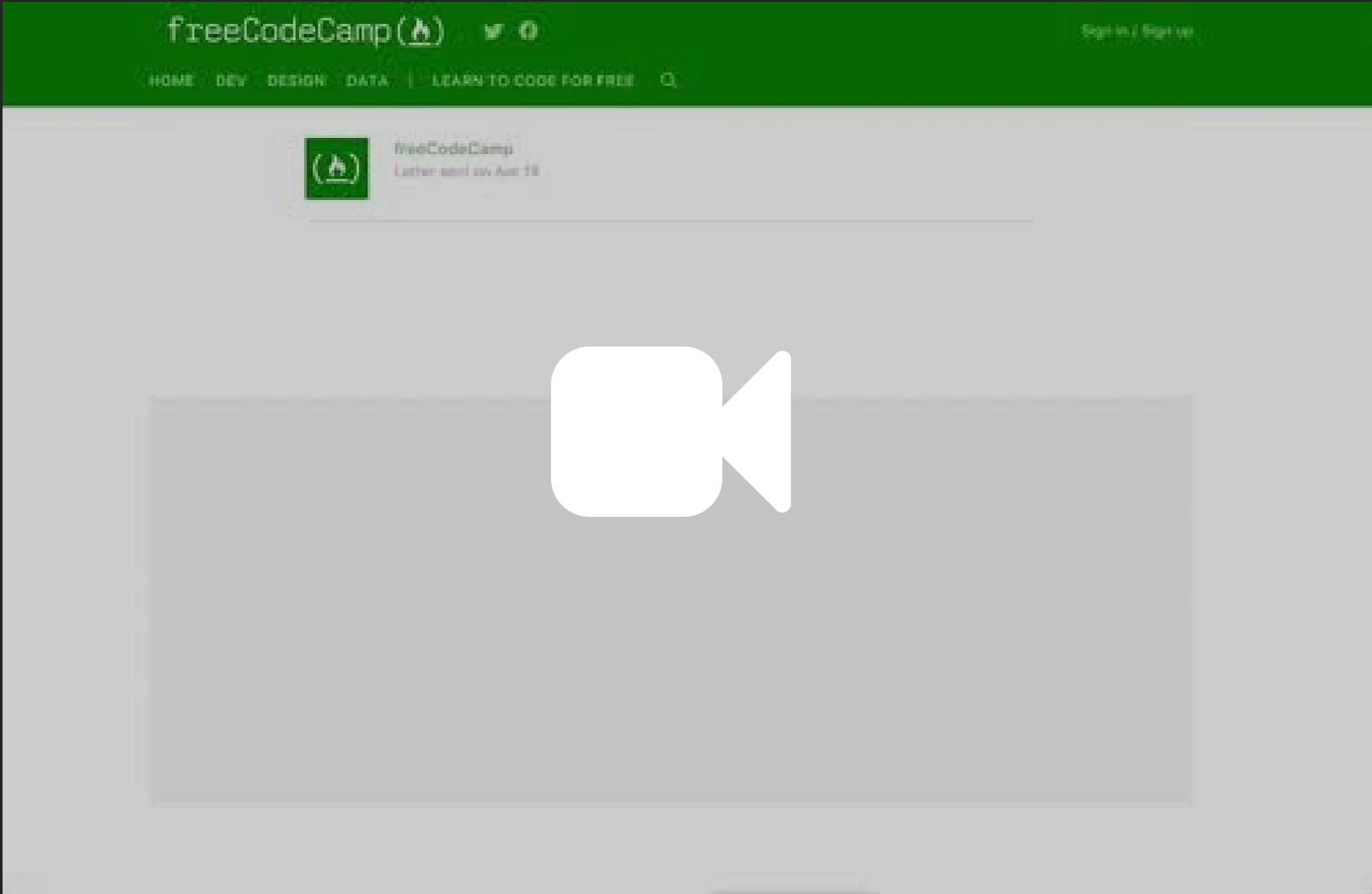
# Ejemplos de carga progresiva de imágenes

# Medium

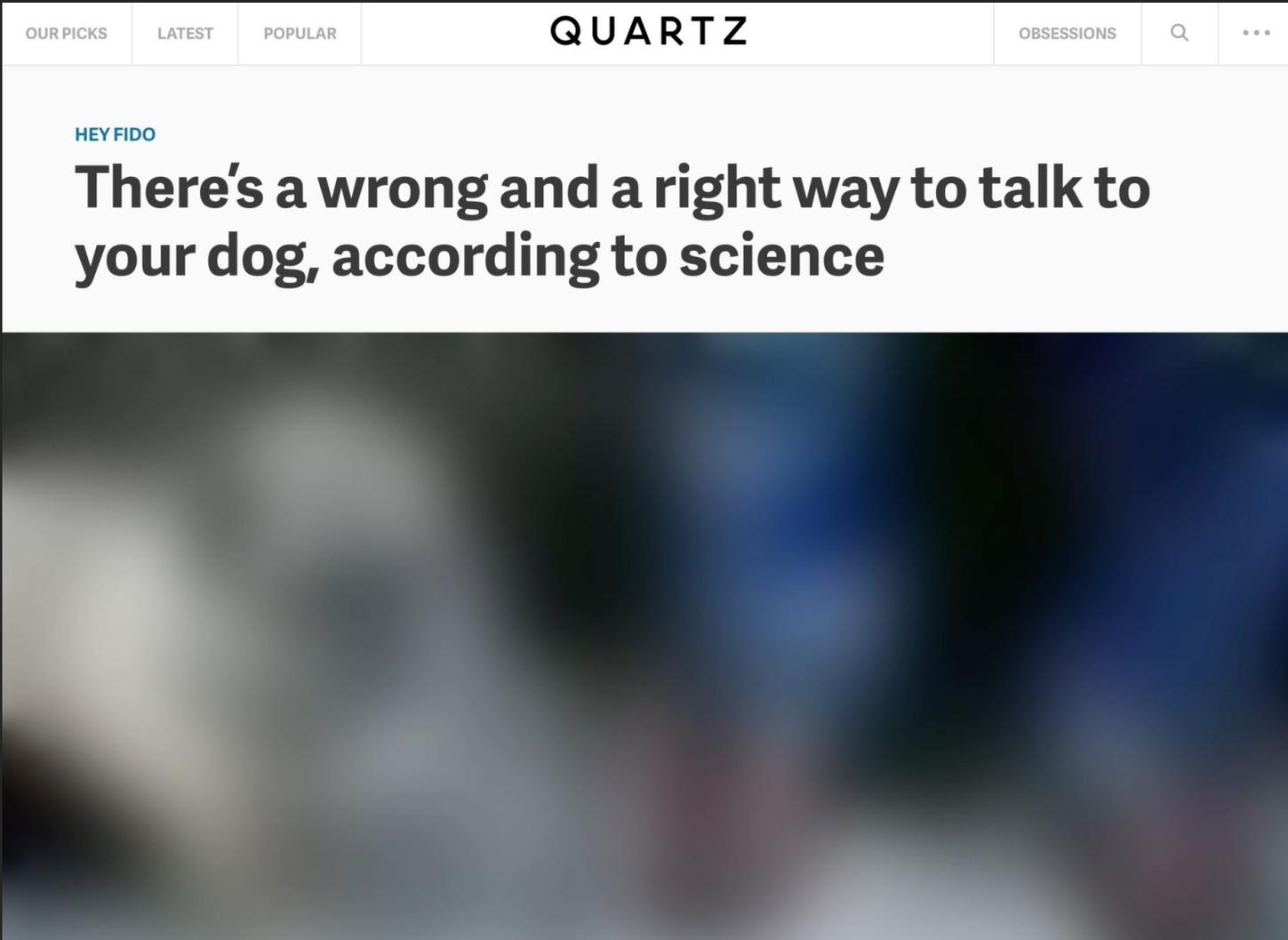


The image shows a screenshot of a Medium article page. At the top, there is a green navigation bar with the freeCodeCamp logo (a flame in a circle) and social media icons for Twitter and Facebook. To the right of the logo, it says "Sign in / Sign up". Below the navigation bar, there is a white header area with the text "HOME DEV DESIGN DATA | LEARN TO CODE FOR FREE" and a search icon. The main content area is white and features the freeCodeCamp profile picture (a green square with a white flame icon) and the text "freeCodeCamp Letter sent on Apr 19". Below this, the article title "A roadmap to becoming a web developer in 2017" is displayed in a large, bold, black font. Underneath the title is a large, blurred image that appears to be a chart or diagram. At the bottom of the page, there is a caption: "These charts were created by GitHub user Kamranahmedse. They visualize".

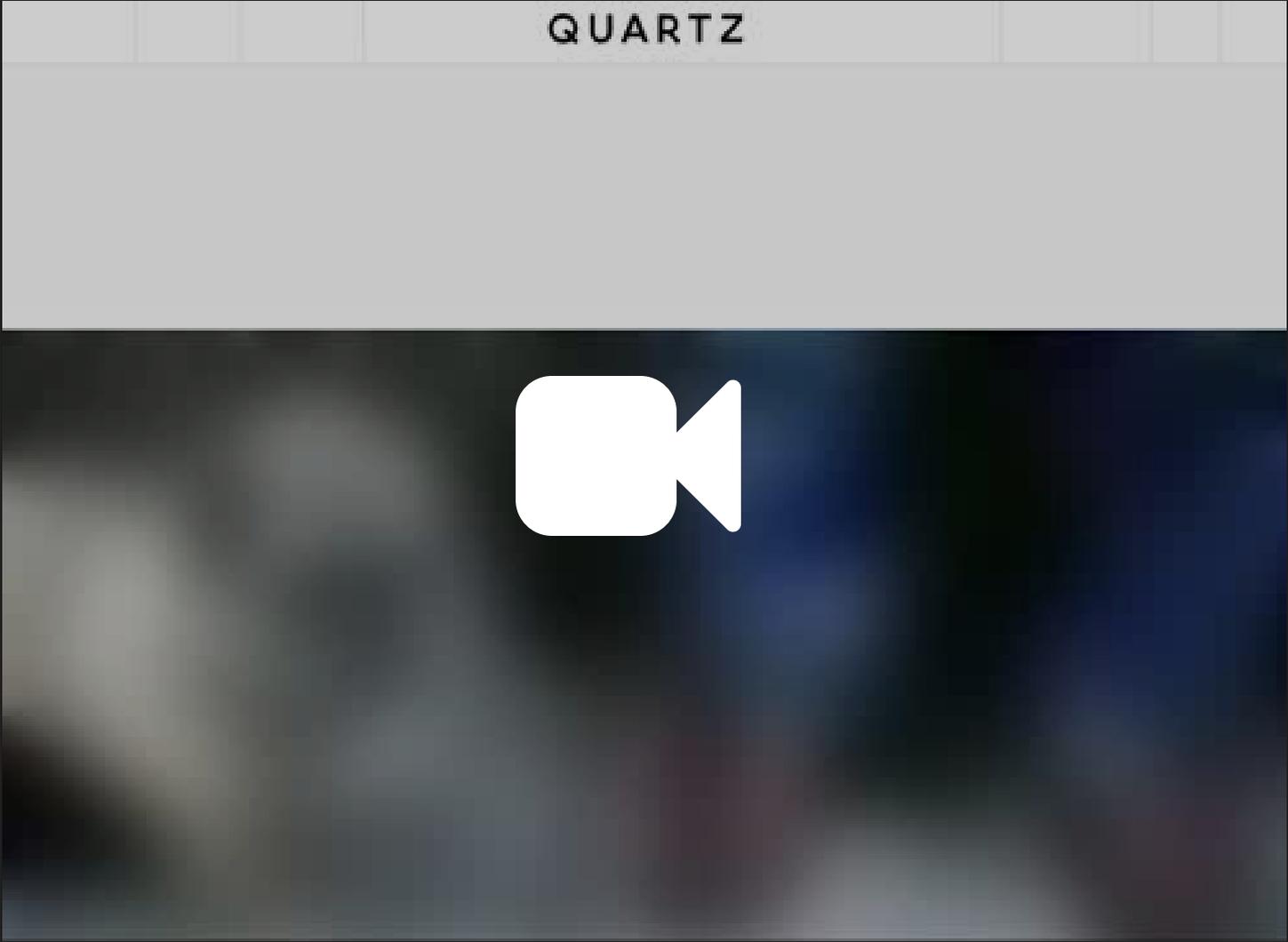
# Medium



# Quartz (qz.com)



# Quartz (qz.com)



# Quora

**Quora** Ask or Search Quora Ask Question

Fast Web Media Quora Product Development +4

## How does Quora (pre) load images?

I thought it's cool how Quora preloads the main color of images, but when I simulated a slow Internet connection in F12 tools I actually saw that it's not just the main color but already part of the texture is already there. Which technology does Quora use to achieve that effect?



Answer Request Follow **13** Comment Share Downvote ...

# Cómo se hace



# HTML - Medium

```
<figure>
  <div>

    <div/> <!-- este div mantiene el espacio con la correcta
           proporción de la imagen, para que no se "colapse" -->

    <img/> <!-- pequeña imagen con baja resolución (por
           ejemplo 27x17) y baja calidad -->

    <canvas/> <!-- toma la imagen anterior y aplica filtro "blur" -->

    <img/> <!-- la imagen final (grande) que se va a mostrar -->

    <noscript/> <!-- fallback para cuando no hay JS -->

  </div>
</figure>
```

# JPEGs progresivos



Baseline



Progressive

***“ Con el método de JPEGs progresivos [...] la fluidez cognitiva se inhibe y el cerebro tiene que hacer un mayor esfuerzo para darle sentido a lo que se está mostrando.***

— De *Progressive image rendering: Good or evil?*

# Thumbnails

## JPG



464 B



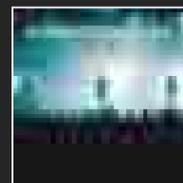
532 B



428 B



409 B

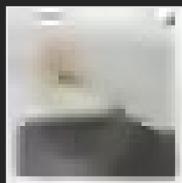


456 B



692 B

## WebP



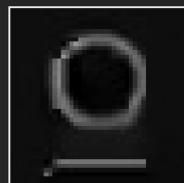
112 B



154 B



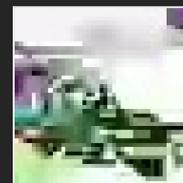
106 B



96 B



116 B



202 B

42 x 42px con máxima compresión

Fuente: <https://jmperezperez.com/webp-placeholder-images/>

# Siendo creativos con SVGs

# SVG



# SVG



# Dibujando con SVG

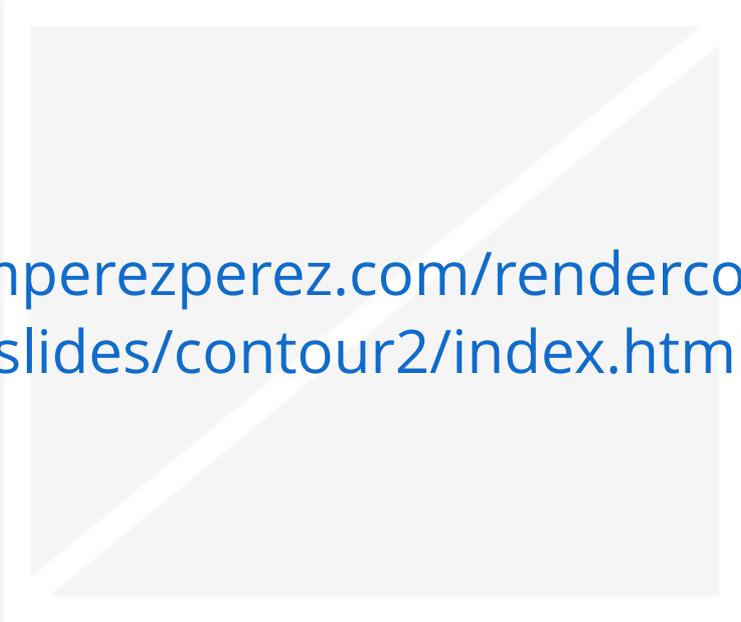


```
">//codepen.io/jmperez/embed/rxxRRg/?height=525&theme-  
id=0&default-tab=js,result&embed-version=2">
```

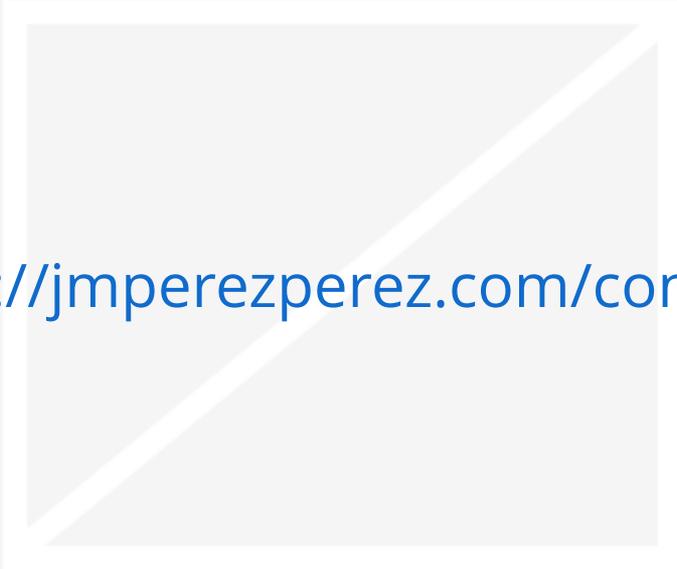
# Dibujando imágenes de mapa de bits



Canny Edge Detector



<https://jmperezperez.com/renderconf17/ext-slides/contour2/index.html>



<https://jmperezperez.com/contour/>

# Cómo dibujar imágenes

1. Buscar el contorno con un detector de bordes Canny
2. Crear líneas
3. Usar JS y SVG para animar

```
<svg>
  <polyline points="51,1 61,1 61,2 56,4 56,3"/>
  <polyline points="52,1 50,2 51,3 50,4 50,9 46,10 46,8 48,8 48,9"/>
  <polyline points="61,4 61,5 58,6"/>
  ...
  <polyline points="62,58 61,59 61,60 50,62 50,61 51,61"/>
</svg>
```

# ¿Deberíamos hacer esto?

Que se pueda hacer no significa que se deba hacer

# Resumen

- Reducir peticiones
- Elegir el formato adecuado y optimizar
- Utilizar imágenes responsive
- Aplicar lazy loading
- ¡Innovar!

**Desarrollar Web es  
divertido.**

[spotify.com/jobs](https://spotify.com/jobs)

buscamos desarrolladores web,  
preguntadme para saber más

iGracias!

@jmperezperez