

## Cadriel Web côté client

Cours 11 - Médias (photos et vidéos)

Steve Lévesque, Tous droits réservés © où applicables

# Table des matières

- 1 Chargement des images et des vidéos (non protégées)
- 2 Chargement des images (non protégées)
- 3 Importation du module Next.js pour optimiser automatiquement les images
- 4 Reduction du poids des images pour réduire le temps de chargement
  - Évaluation de performance
  - Nouvelles approches - WebP
- 5 Mettre des liens de vidéos hébergées de l'externe (non protégées)

# Chargement des images et des vidéos (non protégées)

Il y a des méthodes spécifiques pour ajouter des images et des vidéos dans un site Web avec Next.js.

NB : Nous allons voir dans le prochain cours : Cadre Web côté serveur, comment utiliser des ressources sécurisées. Pour le moment, nous ne pouvons pas puisque **les variables d'environnement sont public**.

# Chargement des images (non protégées)

Le chargement des images est en deux parties :

- L'importation et utilisation du module next/image pour optimiser l'utilisation des images
- Réduction du poids des images avec un logiciel simple pour améliorer la rapidité de chargement

# Import du module Next.js pour optimiser automatiquement les images

Il suffit d'importer et utiliser le module `next/image` pour avoir les avantages suivants :

- **Optimisation de la taille** : servez automatiquement des images correctement dimensionnées pour chaque appareil, en utilisant des formats d'image modernes tels que WebP et AVIF.
- **Stabilité visuelle** : empêchez automatiquement le déplacement de la mise en page lors du chargement des images.
- **Chargements de page plus rapides** : les images ne sont chargées que lorsqu'elles entrent dans la fenêtre d'affichage à l'aide d'un chargement paresseux du navigateur natif, avec des espaces réservés de flou en option.
- **Flexibilité des artifacts (images)** : redimensionnement d'image à la demande, même pour les images stockées sur des serveurs distants.

# Import du module Next.js pour optimiser automatiquement les images

Attention : aucun import ne peut être asynchrone pour pouvoir être lu au “build time” directement.

```
import Image from 'next/image'
import profilePic from './me.png'

export default function Page() {
  return (
    <Image
      src={profilePic}
      alt="Picture of the author"
      // width={500} automatically provided
      // height={500} automatically provided
      // blurDataURL="data:..." automatically provided
      // placeholder="blur" // Optional blur-up while loading
    />
  )
}
```

# Reduction du poids des images pour réduire le temps de chargement

Bien que next/image optimise bien les images, il est difficile tout de même pour un site de charger, par exemple, 100 images haute définition qui pèsent disons 5 MB chacune.

Pourquoi ?

# Reduction du poids des images pour réduire le temps de chargement

Parce que la quantité de données à télécharger est **trop volumineuse**.

Le site sera dans le pire cas figé pendant quelques minutes, sinon tout va être en chargement et les Spinners/loaders vont restés sur le site Web...

Comment peut-on changer la taille des images ?

# Reduction du poids des images pour réduire le temps de chargement

On peut utiliser plusieurs logiciels, payants (Photoshop, etc.) ou gratuits (Gimp, etc.).

Par contre, nous allons utiliser celui dans Windows 11 nommé : Photos (ou Pictures en Anglais).

# Reduction du poids des images pour réduire le temps de chargement

Voici un site d'images gratuites libre de droits :

<https://unsplash.com/s/photos/programming>.

Vous pouvez en télécharger une de votre choix, elle devrait faire 1 MB ou plus puisque la définition est de qualité.

# Reduction du poids des images pour réduire le temps de chargement

Une seule image de 1 MB+ est beaucoup trop lourd pour un site Web, les personnes utilisatrices n'ont pas tous un Internet haute vitesse avec eux au moment du visionnement de votre page.

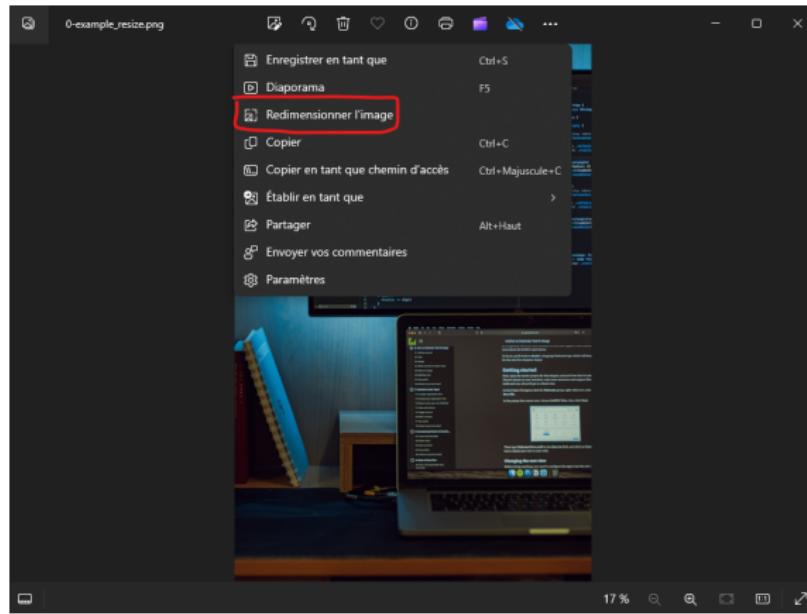
De plus, les appareils utilisées peuvent ne pas supporter la haute définition de votre image, ce qui rend son utilisation inutile.

Nous pouvons réduire la qualité pour ainsi avoir une taille très raisonnable pour le chargement rapide de la page Web.



# Reduction du poids des images pour réduire le temps de chargement

NB : l'interface pourrait avoir changé à cause d'une MAJ.



# Reduction du poids des images pour réduire le temps de chargement

Il est possible de réduire le poids (mémoire) de l'image en changeant la taille et la qualité, vous êtes invités à essayer de réduire d'au moins de x10 la taille tout en gardant un visuel attrayant.

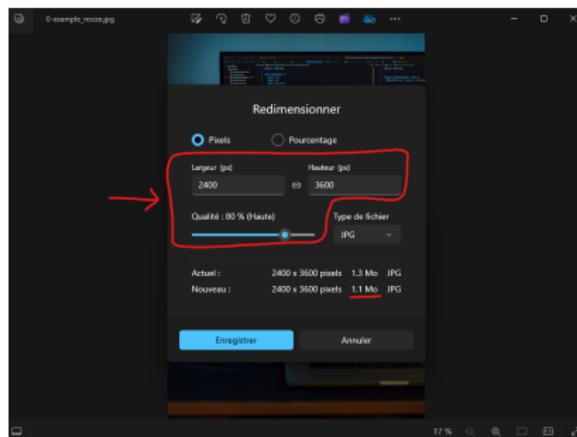


Figure: Avant l'optimisation

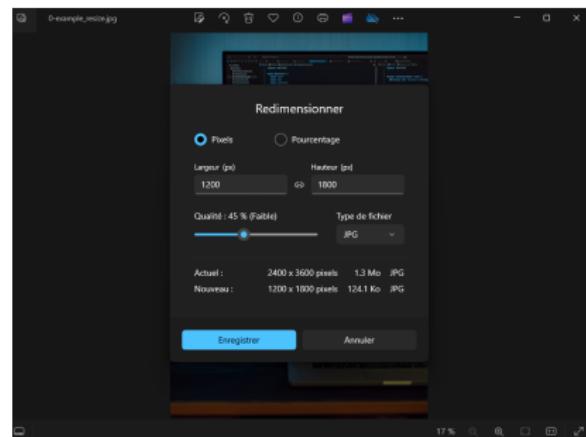


Figure: Après l'optimisation

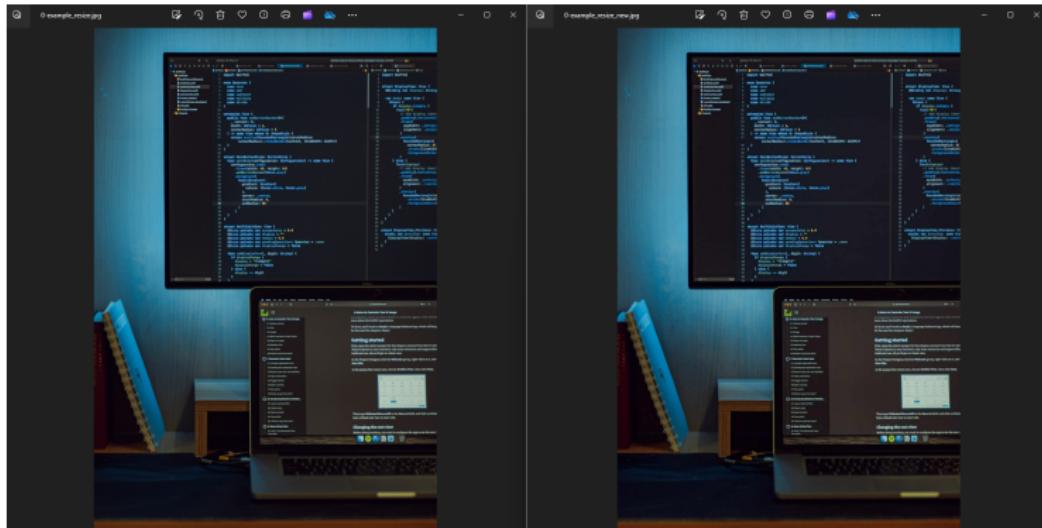
# Reduction du poids des images pour réduire le temps de chargement

Résultat final :

 0-example_resize.jpg	6/16/2024 3:29 PM	JPG File	1,342 KB
 0-example_resize_new.jpg	6/16/2024 3:51 PM	JPG File	125 KB

# Reduction du poids des images pour réduire le temps de chargement - Différence de qualité

Sur le site Web, l'image va paraître quasi-égale pour x10 le poids réduit !



# Reduction du poids des images pour réduire le temps de chargement - Différence de qualité

Bien sûr, en zoomant, nous pouvons voir les défauts, mais ce n'est pas si pire pour le gain de chargement. Il en est primordial pour retenir les utilisateurs sur notre plateforme.

```
34
35
36
37 }
38
39 struct SwiftCalcView: View {
40     @State private var accumulator = 0.0
41     @State private var display = ""
42     @State private var memory = 0.0
43     @State private var pendingOperation: Operation?
44     @State private var displayChange = false
45
46     func addDisplayText(_ digit: String) {
47         if displayChange {
48             display = "\((digit))"
49             displayChange = false
50         } else {
51             display += digit
52         }
53     }
54 }
```

```
34
35
36
37 }
38
39 struct SwiftCalcView: View {
40     @State private var accumulator = 0.0
41     @State private var display = ""
42     @State private var memory = 0.0
43     @State private var pendingOperation: Operation?
44     @State private var displayChange = false
45
46     func addDisplayText(_ digit: String) {
47         if displayChange {
48             display = "\((digit))"
49             displayChange = false
50         } else {
51             display += digit
52         }
53     }
54 }
```

# Évaluation de performance

## Benchmark : simulé sur un réseau 3G Rapide

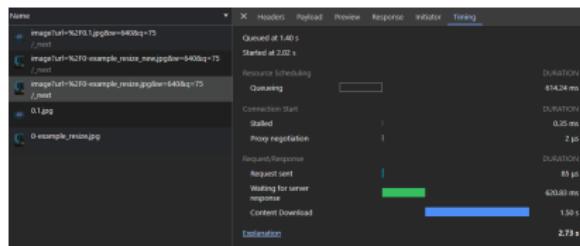


Figure: Image d'origine

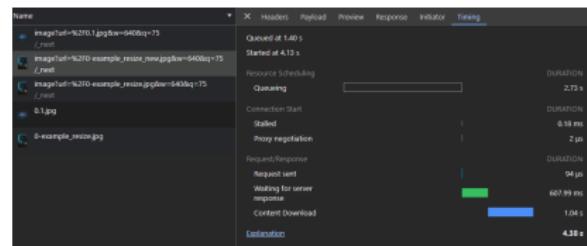


Figure: Image avec réduction de poids

# Évaluation de performance

## Benchmark : simulé sur un réseau 3G Rapide

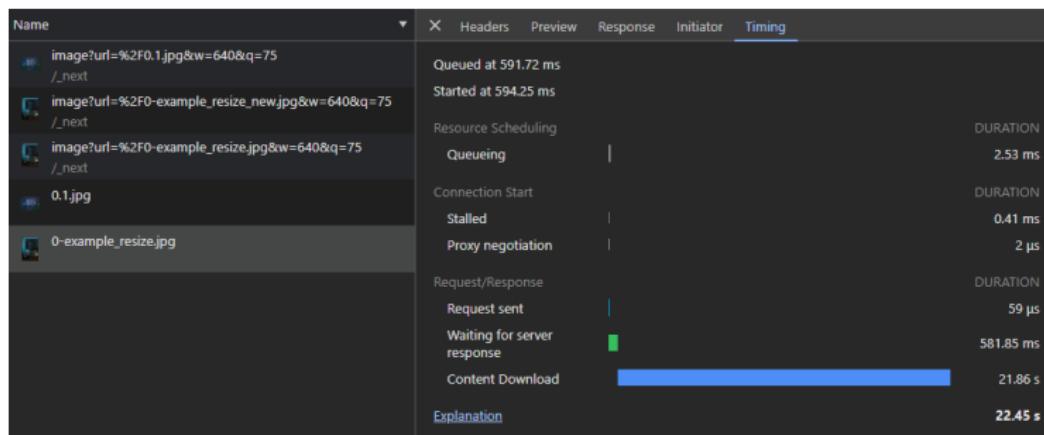


Figure: Image avec le tag traditionnel <img/>

## Nouvelles approches - WebP

Une nouvelle méthode est généralement utilisée : WebP.

<https://en.wikipedia.org/wiki/WebP>

[https://nextjs.org/docs/pages/  
building-your-application/optimizing/images](https://nextjs.org/docs/pages/building-your-application/optimizing/images)

# Mettre des liens de vidéos hébergées de l'externe (non protégées)

Il existe deux balises :

- **<video>** : Permet de jouer des vidéos locales ou d'un serveur de la même famille de domaine du url du site.
- **<iframe>** : Permet d'inclure un site Web externe dans notre site Web, entre autre pour jouer des vidéos Youtube ou de d'autres fournisseurs similaires.

Et des modules quelconques :

- **<ReactPlayer>** : Module convivial pour jouer des vidéos d'un hébergeur (i.e. Youtube, AWS S3, etc.).

# Bibliographie

- [https://nextjs.org/docs/app/  
building-your-application/optimizing/images](https://nextjs.org/docs/app/building-your-application/optimizing/images)
- [https://nextjs.org/docs/app/  
building-your-application/optimizing/videos](https://nextjs.org/docs/app/building-your-application/optimizing/videos)