



# Projet Fish📷ye

Créez un site accessible pour  
une plateforme de photographes



# Contexte du projet

Intégration récente au sein de la société Techasite, société de conseil spécialisée dans le développement de sites web et d'applications mobiles.

Le projet FishEye est un site web qui permet aux photographes indépendants de présenter leurs meilleurs travaux. Ils ont souhaité mettre à jour leur site web.

Je suis chargée de fournir le HTML, CSS et Javascript pour la construction du prototype, en tenant compte du cahier des charges.

# Outils

- Notes de réunion sous forme de cahier des charges

<https://s3.eu-west-1.amazonaws.com/course.oc-static.com/projects/Front-End+V2/P5+Javascript+%26+Accessibility/Notes+de+r%C3%A9union.pdf>

- Maquettes Web :

<https://www.figma.com/file/pt8xJxC1QffW4HX16QhGZJ/UI-Design-FishEye-FR?node-id=120%3A1055>

- Les médias et fichier JSON

# Présentation : index.html

- Maquette :

(typo DM Sans, taille, couleurs)

- logo
- nav
- titre
- photographes
- tags

Fish📷ye

#portrait #events #travel #animals #sport #architecture #art #fashion

Nos photographes



Mimi Keel

London, UK

Voir le beau dans le quotidien  
800€/jour

#portrait #events #travel #animals



Ellie-Rose Wilkens

Paris, France

Capturer des compositions complexes  
250€/jour

#sport #architecture



Tracy Galindo

Montreal, Canada

Photographe freelance  
100€/jour

#art #fashion #events



Nabeel Bradford

Mexico City, Mexico

Toujours aller de l'avant  
350€/jour

#travel #portrait



Rhode Dubois

Barcelona, Spain

Je crée des souvenirs  
270€/jour

#sport #fashion #events #animals



Marcel Nikolic

Berlin, Germany

Toujours à la recherche de LA photo  
300€/jour

#travel #architecture

# Construction : index.html

- intégrer les données des photographes JSON dans le Javascript :
  - Méthode `fetch()` : effectuer une requête Get et obtenir nos ressources.
  - Méthode `then()` : introduire les fonctions synchrones et asynchrones pour le fonctionnement de Javascript. ( renvoie une promesse ) ( **voir *index.js*** )

```
fetch('data.json') // renvoie une promesse
.then((response) => response.json()) // renvoie aussi une promesse
.then((data) => {
    let list = new List();
    //-- afficher les 6 photographes (Photographer.js)
    data.photographers.forEach((item) => {
        let photographe = new Photographer(item);
```

# Construction : index.html

- POO : programmation orientée objet

employé une construction des éléments en utilisant un constructeur dans une classe, qui permet d'instaurer les propriétés que l'on souhaite.

la construction des éléments est faite par le javascript, moins de balises dans le html

- render ( voir photographer.js)
- fonctionnalité ( voir list.js)
- les méthodes sont appelées dans l'index.js
- les fichiers reliés au html

```
class Photographer
{
  constructor(data)
  {
    this.name= data.name;
    this.id= data.id;
    this.city= data.city;
    this.country= data.country;
    this.tags= data.tags;
    this.tagline= data.tagline;
    this.price= data.price;
    this.portrait= data.portrait;
    this.alt = data.alt;
  }
}
```

# Fonctionnalités : Tags

## Tags navigation :

- récupérer toute la liste des tags : **getTags()** (voir **list.js**)
- afficher tags dans la navigation : **displayTag ()**
- utiliser la méthode **filter()** pour récupérer les données à partir d'un tableau créé. **filter()**
- utiliser l'objet **Set()** pour éviter les doublons
- listener pour filtrer les éléments au click : **listenForFilter()**

## Tags photographes :

- même principe sauf qu'on filtre avec **listenForFilterTags()**

#portrait

#events

#travel

#animals

#sport

#architecture

#art

#fashion

# Présentation : photographers.html

## - Maquette :

- logo
- présentation du photographe + tag
- formulaire
- dropdown
- médias
- compteur likes + prix

Fish@ye


### Mimi Keel

London, UK

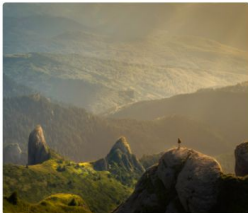
Voir le beau dans le quotidien

#portrait #events #travel #animals


Contactez-moi




Trier par Popularité




Lonesome 45€ 88♥





Hillside Color 45€ 85♥



Wednesday Portrait 45€ 34♥







680 ♥ 400€/jour



# Construction: photographers.html

- Relier la page photographers.html à la page index.html par une chaîne de requête URL

```
//-- chaîne de requêtes url pour récupérer les pages photographes en fonction de leurs ID  
const url = new URLSearchParams(window.location.search);  
const id = url.get('id');
```

- Utiliser l'api fetch() :
  - lancer une requête de type Get et obtenir nos ressources. (**profil.js**)
- Utiliser méthode then() :
  - introduire les fonctions synchrones et asynchrones. (renvoi promesse) (*voir profil.js*)
- Utiliser la méthode find():
  - permet d'obtenir l'élément qu'on veut.

# Construction: photographers.html

- Photographes : **displayProfile()** + son render
  - Dropdown : **displayFilter()** + render
  - Total likes + Info : **displayInfo()** + render
- 
- Tags de présentation reliés au tags navigation
  - Tags filtrent les portraits en relation avec la page index.html

London, UK

Voir le beau dans le quotidien

#portrait

#events

#travel

#animals

# Fonctionnalité : Media Factory

Factory Pattern permet de :

- déléguer la création d'objet tout en gardant le contrôle du type de données à instancer
- être extensible
- obtenir une approche solide pour la résolution de problème
- rendre le code plus lisible grâce aux modèles réutilisables
- utiliser un modèle "constructor"

```
class MediaFactory
{
    build(data)
    {
        //permet d'insérer tous types de médias
        if (data.hasOwnProperty('video'))
        {
            return new Video(data);
        }

        if (data.hasOwnProperty('image'))
        {
            return new Image(data);
        }
    }
}
```

# Media Factory(2)

- POO et construction sur le media Factory
  - récupérer la source médias.
- Hydrater le code
  - récupérer le build des médias factory: hydrate() ( ***voir portfolio.js***)
- Formater les données JSON
  - créer une classe média avec son constructor ( ***voir media.js***)
- Travailler sur les images et les vidéos

# Fonctionnalité : Dropdown

- Format Desktop
- Ouvert par la flèche : **listenForDropDownToggle()**

Popularité ▼

- Filtre sur les médias par : Popularité / Date / Titre : **listenForFilters()**

Popularité ▲

Date

Titre

# Fonctionnalité : Likes

- Créer compteur des likes +/- ( ***voir media.js*** )
- Créer compteur total : creatCount() ( ***voir portfolio.js*** )
- Incrémenter / décrémenter : **listenForReactions()**



Seaside Wedding

45€ 26♥

681 ♥

400€/jour

# Fonctionnalité : Ligthbox

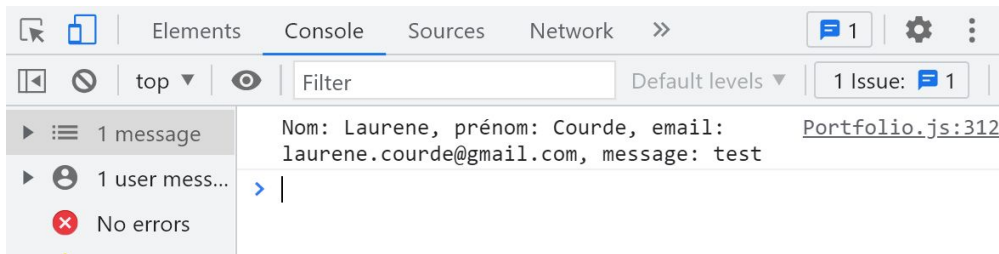
- lancer le slider : **listenForSlider()**
- boutons : close/next/Previous
- ordre opérationnel en fonction du filtre



Benevides Wedding

# Fonctionnalité : Formulaire

- modale de lancement et de fermeture
- saisie des "inputs"
- validation avec données dans les logs



×

## Contactez-moi

### Mimi Keel

Prénom

Nom

Email

Votre message

Envoyer




# Responsive

**Fish@ye**


**Ellie-Rose Wilkens**

Paris, France  
Capturer des compositions complexes


#sport #architecture



Tricks in the Air 70€ 150 ♥



Climber 65€ 101 ♥



Surfer 70€ 103 ♥

Contactez-moi

**Fish@ye**


**Ellie-Rose Wilkens**

Paris, France  
Capturer des compositions complexes


#sport #architecture

Contactez-moi


Trier par Popularité ▼



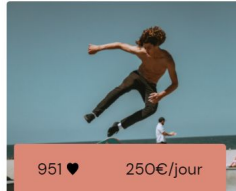

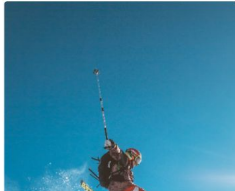
Tricks in the Air 70€ 150 ♥



Climber 65€ 101 ♥



Surfer 70€ 103 ♥



951 ♥ 250€/jour

# Accessibilité

Mise à disposition des sites web au plus grand nombre d'utilisateurs :

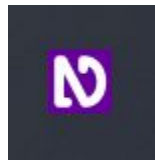
- personnes ayant un handicap
- les utilisations d'appareils mobiles
- connexion internet à faible débit

**Considérer tout le monde de la même manière et donner les mêmes opportunités à tous, peu importe leurs handicaps ou les circonstances.**

- Balises sémantiques HTML5 ( balise alt pour les images - hiérarchie des titres H1 à H6, etc... )
- Attributs ARIA qui complètent les balises HTML quand celles-ci sont insuffisantes et non explicites
- Les rôles des éléments

# Accessibilité : présentation

- Présenter site Web avec le clavier sur une page
- Présenter site Web avec le lecteur d'écran NVDA sur une autre page



# Techniques

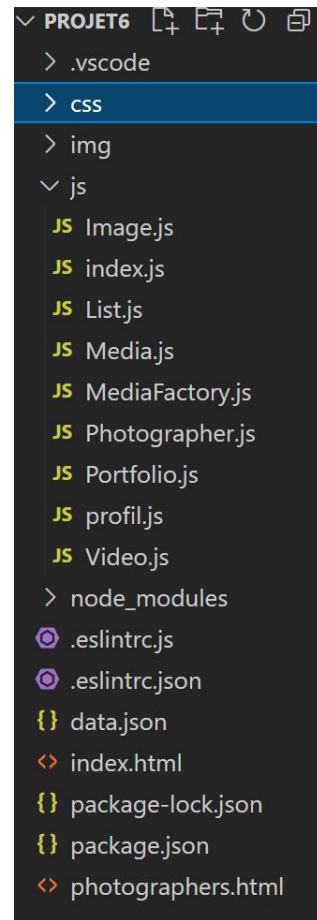
- Organisation de fichiers par dossiers
- Eslintrc.js

Lien Github:

[https://github.com/Laurene45/LaureneCourde\\_6\\_05072021](https://github.com/Laurene45/LaureneCourde_6_05072021)

Lien GithubPages:

[https://laurene45.github.io/LaureneCourde\\_6\\_05072021/](https://laurene45.github.io/LaureneCourde_6_05072021/)



# Validation W3C

- Html

**Document checking completed. No errors or warnings to show.**

Used the HTML parser.

- Css

Résultats de la validation W3C CSS de contact.css (CSS niveau 3 + SVG)

**Félicitations ! Aucune erreur trouvée.**

- Achecker

**Known Problems(0) Likely Problems (0)**

 **Congratulations! No known problems.**

# Avez-vous des questions ?

