

Développement d'une interface utilisateur web

Projet n°6 du parcours python d'openclassrooms

Bérenger Ossété Gombé

19 juillet 2022

Sommaire

- 1 Introduction
- 2 Préparation du projet
- 3 Démonstration
- 4 Mise en place du *front-end*
 - HTML
 - CSS
 - Javascript
- 5 Conclusion

Présentation

Je me présente

- Béranger Ossété Gombé, 27 ans
- Mon parcours informatique
 - Baccalauréat scientifique (2013)
 - Master 1 en informatique à l'UFC, UFR-ST (2016-2017)
 - Spécialité génie logiciel.
 - Reconversion vers du développement web chez Openclassrooms (janvier 2022).

Formation

Contexte

Projet n°6 du parcours Développeur d'application python chez openclassrooms.

Objectifs

- Développer la partie *front-end* d'une application web.
 - HTML
 - CSS
 - Javascript
- Interagir avec une API REST.

Contexte fictif

JustStreamIt

- Association de cinéphiles.
- Système de *newsletters*.
- Migration vers un site web.
- Équipe
 - Camille, la CEO de JustStreamIt.
 - Éric, notre contact technique.

Notre travail

→ Nous sommes volontaire pour développer le *front-end* du nouveau site. Nous utiliserons leur API existante.

Préparation du projet

Étapes

- Collecte des exigences du client.
- Tests et documentation.
- Normes et qualité.

Exigences fonctionnelles

Une liste des exigences fonctionnelles

- Conception **conforme à la maquette** comprenant :
 - une barre de navigation,
 - une vignette présentant le meilleur film,
 - une catégorie des films les mieux notés,
 - trois catégories aux choix comprenant sept films.
- **Informations d'un film** *via* une fenêtre modale contenant :
 - l'image, le titre, le genre, la date de sortie,
 - le classement, le score IMDB,
 - le réalisateur, les acteurs,
 - la durée, la pays d'origine, le résultat au *box office* et
 - le résumé du film.

Exigences non-fonctionnelles

Une liste des exigences non-fonctionnelles

- **Portabilité** vers les navigateurs Chrome, Safari et Firefox.
- Utilisation de l'API **OCMovies-API** et accès *via* des requêtes AJAX.
- **Mise-à-jour automatique** des données.
- Utilisation de Javascript et de CSS *sans **framework** ni bibliothèque.*

Tester le *front-end*

Pourquoi tester le *front-end* ?

- Vérifier que le produit est conforme aux exigences.
- Mettre en place des vérifications pour accompagner une potentielle future mise à l'échelle.
- Documenter les bugs et éviter autant que possible les régressions.

Tester le *front-end*

Utilisation de Jasmine

Choix d'utiliser Jasmine car¹ :

- Possède peu de dépendances.
- Est un projet très stable.
- Une utilisation simple est possible.
- → Prend peu de place au sein du projet.

1. Voir

<https://npmcompare.com/compare/chai,jasmine,jest,karma,mocha>

Documenter le *front-end*

Pourquoi documenter le *front-end* ?

- Le projet peut être repris à l'avenir par quelqu'un d'autre.
- Le projet peut grossir et se complexifier.

Utilisation de JSDoc

Choix d'utiliser JSDoc car :

- facile et rapide à installer,
- facile à utiliser,
- génère une documentation claire et concise.

Normes et qualité

Vérifier le code HTML

- Permet de détecter les erreurs dans le code HTML.
- Permet de garantir la conformité du code avec les recommandations de la W3C².
- Disponible sous forme d'extension pour navigateurs³.

2. World Wide Web Consortium

3. Disponible au moins pour Chrome et Firefox.

Normes et qualité








Type	Line	Column	HTML errors and warnings
 Info	0	0	0 errors / 23 warnings
 Warning	29	168	Warning: missing </div> before <a>
 Warning	32	4	Warning: discarding unexpected
 Warning	32	14	Warning: discarding unexpected </div>
 Warning	58	184	Warning: missing before <div>
 Warning	58	162	Warning: missing before <div>
 Warning	58	363	Warning: inserting implicit

Figure – HTML Validator : une extension firefox pour vérifier le code HTML

Normes et qualité

Vérifier le code Javascript

Utilisation de ESLint², un *linter* pour Javascript.

- Permet de localiser les erreurs javascript.
- Permet également de localiser les erreurs de style.
- Permet de générer un rapport.

2. <https://eslint.org/>

Normes et qualité

ESLint Report

1 problem (1 error, 0 warnings) - Generated on Sun Jun 12 2022 17:15:30 GMT+0200 (heure d'été d'Europe centrale)

[-] /home/bog/Dev/Tutorials/JSESLint/app.js

1 problem (1 error, 0 warnings)

2:7 **Error** 'x' is assigned a value but never used.

[no-unused-vars](#)

Figure – Exemple de rapport généré par ESLint.

Démonstration du *front-end*

Fonctionnalités clefs

- Le menu principal.
- Le meilleur film.
- La catégorie des films les mieux notés.
- Les 3 catégories au choix.
- L'aperçu d'un film *via* une fenêtre modale.

Fonctionnalités additionnelles

- Adaptation au mobile.

Mise en place du *front-end*

Les aspects du développement *front-end*

- La structure avec HTML.
- Le style avec CSS.
- Le comportement avec le Javascript.

Structure HTML du projet

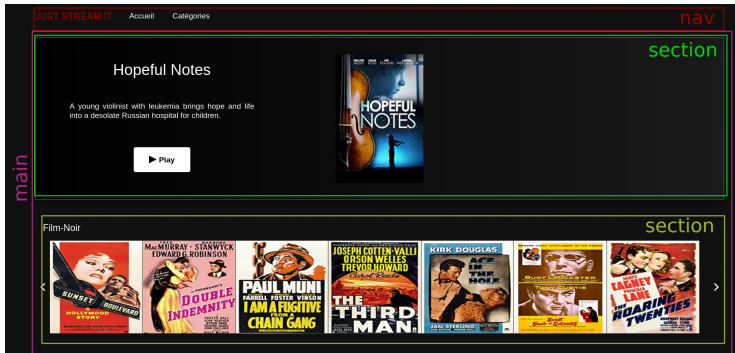


Figure – Éléments HTML de la page d'accueil

Un CSS maintenable et *scalable* avec SASS

Organisation

- Séparation du style sur plusieurs fichiers
- Application du *pattern* 7-1

```
@import 'base/reset';  
@import 'abstracts/mobile';  
@import 'themes/theme';  
@import 'layout/nav';  
@import 'pages/home';  
@import 'components/buttons';  
@import 'components/best_movie';  
@import 'components/movie_info';
```

Un CSS *responsive*

aze

- Approche *Mobile first* lors du développement.
- Utilisation d'une mixin SCSS pour gérer les *media queries*.

```
@mixin for-mobile {  
  @media screen and (max-width: 599px) {  
    @content  
  }  
}
```

Un CSS *responsive* : exemple

```
@include for-mobile {  
    position: fixed;  
    margin: 0;  
    top: 0;  
    left: 0;  
    width: 100%;  
    min-width: 0;  
    height: 100%;  
    max-height: 100%;  
}
```

Un CSS organisé avec la méthode BEM

La méthode BEM

- Permet d'organiser le CSS.
- Extensible au SCSS.
- Signifie *Block* - *Element* - *Modifier*.
- Propose un formatage dans le choix des noms de classe CSS d'un projet.

Un exemple de formatage

nom_*bloc* __ nom_*element* - - nom_*modifier*

Le BEM appliqué au SASS

En SASS

```
.best_movie {}  
  &__title {}  
    &--blue {}  
}
```

En CSS

```
.best_movie {}  
.best_movie best_movie__title {}  
.best_movie best_movie__title best_movie__title--blue {}
```

Structure du code Javascript

Utilisation de NPM

Récupération des informations *via* l'API

Génération HTML des catégories

Mise en place du carrousel

Gestion de la modale

Conclusion

Pour conclure

- Les limites du projet.
- Pour aller plus loin.
- Travail effectué.

Les limites du travail réalisé

Pour aller plus loin

Travail effectué

Merci pour votre attention

- 1 Introduction
- 2 Préparation du projet
- 3 Démonstration
- 4 Mise en place du *front-end*
 - HTML
 - CSS
 - Javascript
- 5 Conclusion