L’atelier de développement web

Les outils de production

Nous allons mettre en place notre environnement de travail. Chaque équipe de dev a ses propres méthodes et outils, il est crucial de savoir s’y adapter. Toutes ces boîtes à outils sont composées au minimum d’un **navigateur** (voir), d’un **éditeur de texte** (modifier) et d’un système de **gestion de version** (sauvegarder).

Remise en forme

Les langages du web

HTML

Anatomie d’une balise :

```
<img src="logo.png" alt="AppChat">
```

- Les angles : **< >** indiquent les balises (ouvrantes ou fermantes **</ >**)
- En orange le nom de la balise
- En rouge les attributs et en blanc leur valeur.
- **OpenClassRoom : memento des balises html**

CSS

Les sélecteurs et propriétés de styles :

```
p {
  font-size: 18px;
}

.chat {
  background: #eff2f5;
}
```

- En orange un sélecteur d’élément, il cible toutes les balises **p**.
- En rouge un sélecteur de classe, il est précédé d’un point, il cible les balises dont l’attribut **class** contient **chat** : `<div class="chat"> … </div>`
- En violet les propriétés, en jaune leur valeur.

- Attention à bien séparer chaque couple propriétés / valeur par des points virgules.
- **OpenClassRoom : memento des propriétés css**

Prise en main de Git
La gestion de version

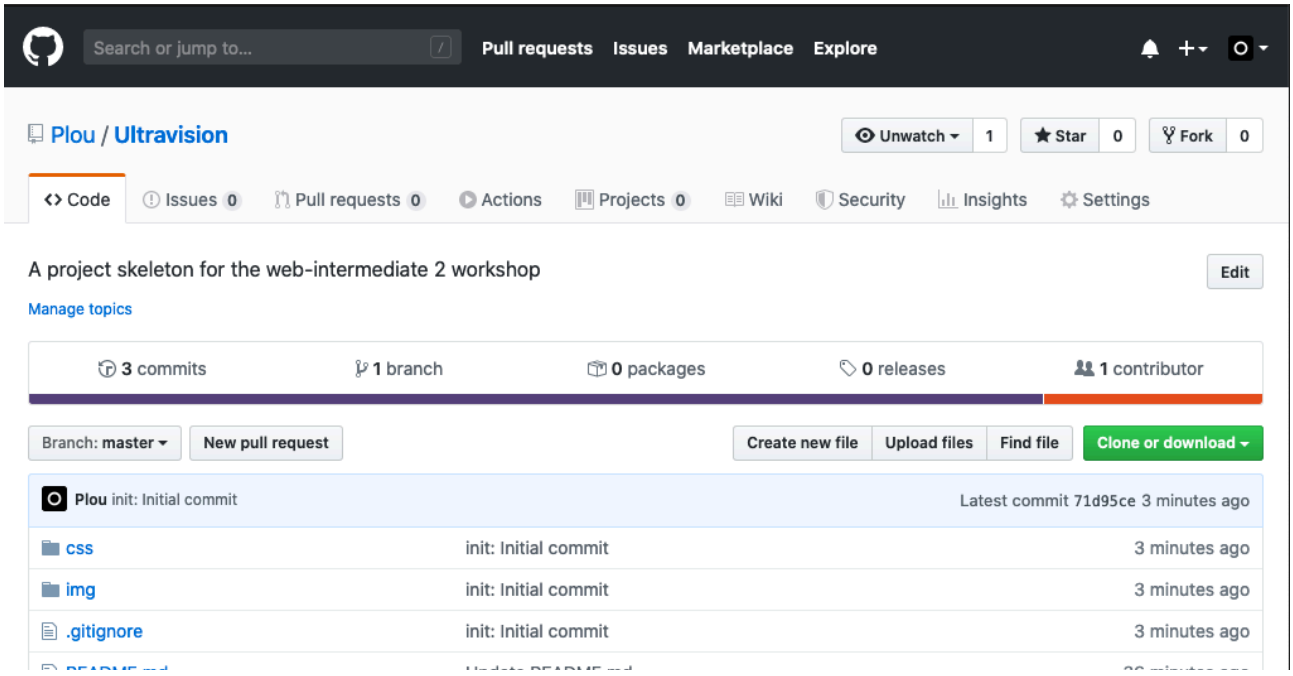
Git est un outil informatique qui permet de gérer les **versions** d'un projet (l'ensemble des fichiers). Il permet de créer des repères (**commit**) à chaque étape de travail (modification). L'historique ainsi créé va nous permettre de comprendre facilement toute les strates de code qui ont aboutis au projet actuel. En plus d'avoir des capacités de machine à voyager dans le temps, il permet de mieux collaborer. Dans le monde de l'open source, c'est l'outil de prédilection pour travailler et publier des applications, site web, livres...

Utilisation de GitHub

GitHub est un service en ligne d'hébergement de dépôts (**repository**). Tout les fichiers et l'historique du projet y sont sauvegardés.

Commençons par en créer une copie.

Une fois connecté·e à votre compte GitHub, rendez-vous sur la page :
https://github.com/Plou/Ultravision
Puis cliquez sur le bouton **Fork**[1] en haut à droite.



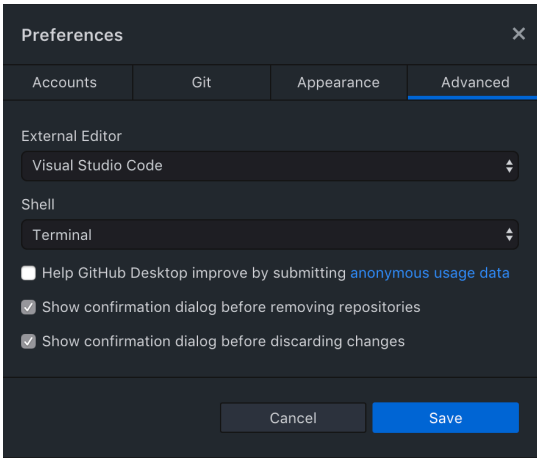
Après quelques secondes, une copie du dépôt sera associée à votre compte. Vous travaillerez sur votre propre version du site.

[1] Fork : https://fr.wikipedia.org/wiki/Fork_(développement_logiciel)

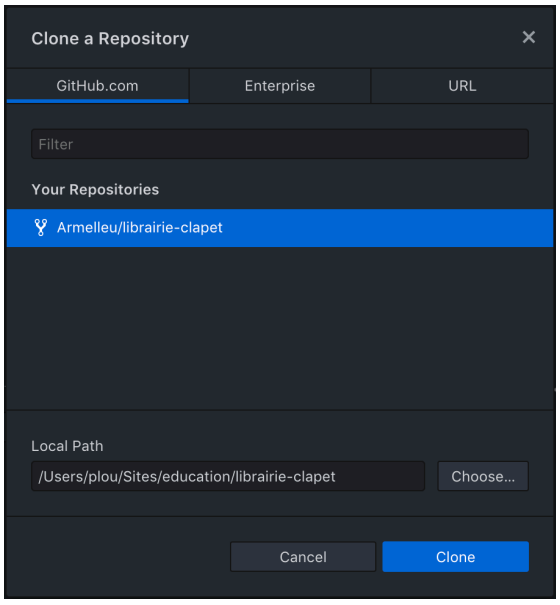
Récupération du projet
Cloner le projet avec GitHub Desktop

La première étape : récupérer les fichiers sur notre machine, nous travaillerons **en local**. Pour simplifier ce processus nous utiliserons **GitHub Desktop**, un logiciel permettant de gérer nos dépôts.

Connectez vous avec votre compte GitHub, puis dans les préférences allez choisir l'éditeur de texte choisi : VSCode, Atom ou Sublime Text

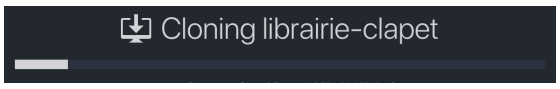


Cliquez ensuite sur **Clone a Repository**, pour récupérerez le dépôt git :



Sélectionnez votre projet dans la liste de vos dépôts (**repositories**).

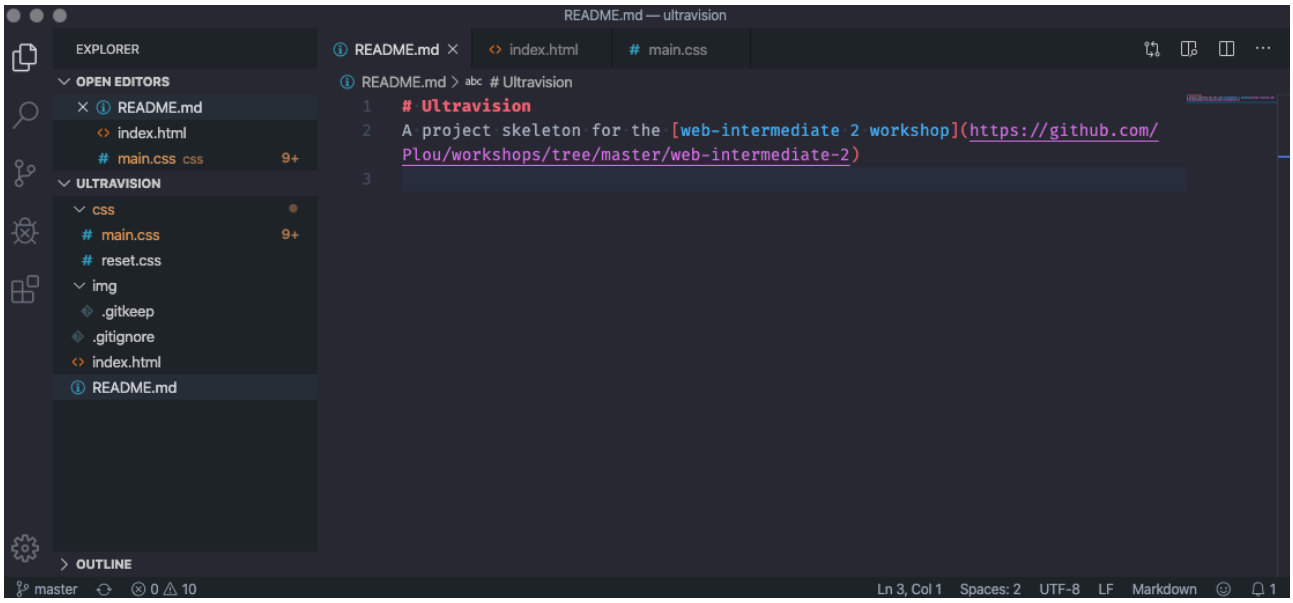
Vous pouvez modifier l'emplacement des fichiers sur votre ordinateur pour être sûr·e de les retrouver.



Validez (**Clone**), le dépsôt sera alors téléchargé sur votre machine.

Exploration et prise en main du projet

Vous avez maintenant tous les fichiers nécessaires pour travailler.
Accédez au dossier via le menu : **Repository > Show in Finder/Explorer**.
Ouvrez le projet avec votre éditeur : **Repository > Open in VSCode**.



Sur la gauche de votre éditeur, vous visualisez l'arborescence des fichiers.
Cliquez dessus pour les afficher et les modifier.

Mises à jours

Modifications

L'interêt de git est de permettre le **versioning**, c'est la capacité à visualiser chaque étape de travail. On parle de **commit**[1].
Pour mieux comprendre, nous allons procéder à quelques modifications :

- Ajoutez une image[2] dans le dossier **img** de votre projet.
- Ouvrez le fichier **index.html** et ajouter y une balise **img** qui affiche cette image.

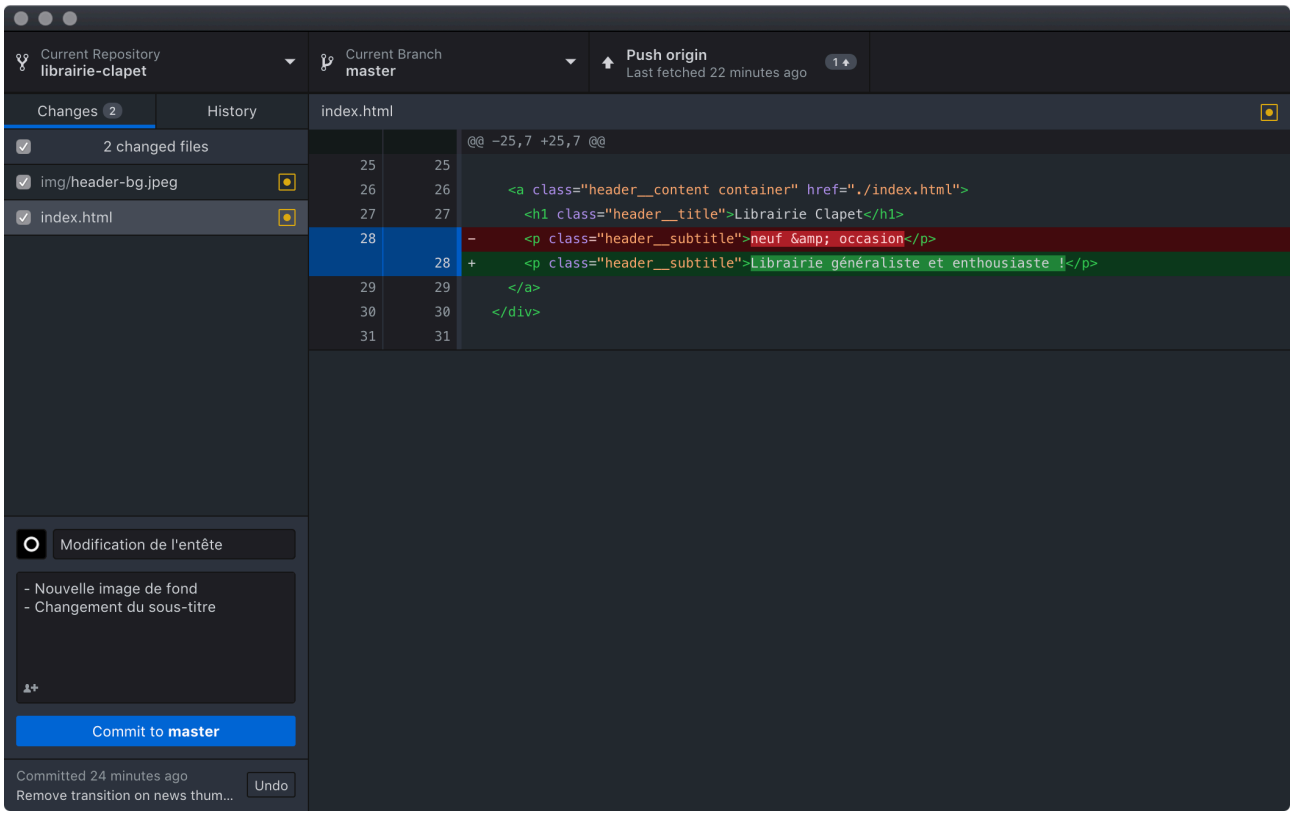
[1] Commit : https://fr.wikipedia.org/wiki/Commit

[2] Banque d'image libres de droit : https://unsplash.com/search/photos/library

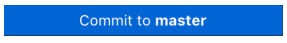
Commit

Une fois satisfait·e·s de vos modifications vous allez pouvoir les ajouter à votre dépôt ce sera l'historique de votre projet.

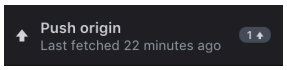
Retournez alors dans GitHub Desktop, vos modifications apparaîtrons :



- À gauche, la liste des fichiers modifiés depuis le dernier commit.
- À droite, un aperçu des modifications.
- En bas à gauche, vous devez **nommer** et **décrire** cette mise à jour.

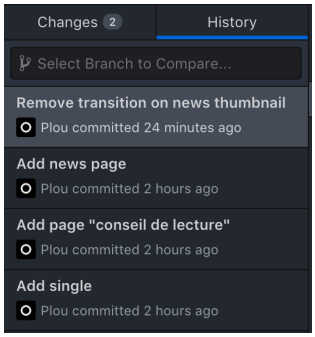


Cliquez sur **Commit to master** pour enregistrer.
Attention ! Cette opération est irréversible.



Pour l'envoyer sur le dépôt principal (**GitHub**). Cliquez sur le bouton **Push origin**.

Ce message de commit viendra s'ajouter à l'historique du projet. C'est cet historique qui vous permet de visualiser l'intégralité de la vie des fichiers et éventuellement revenir en arrière (bug, retrait de fonctionnalités, tests...)[1].
Vous retrouverez également cet historique sur Github.



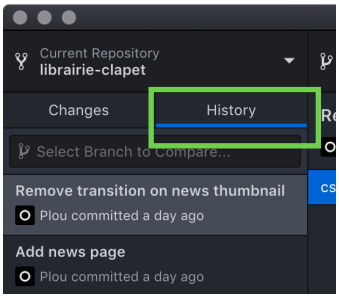
[1] Apprendre git : https://www.codecademy.com/learn/learn-git

Notation

Chaque étape de l'atelier devra faire l'objet d'un **commit** (minimum). Je vous conseille de faire un commit à chaque fois que vous obtenez un résultat correct : ajout d'une nouvelle page, création d'un nouveau composant, changement de taille de polices…

J'utiliserai l'historique des commit pour évaluer votre travail.

Pour le visualiser, rendez vous dans l'onglet commit de GitHub ou L'historique de GitHub Desktop :



https://github.com/Plou/Ultravision/commits/master