

Les versions

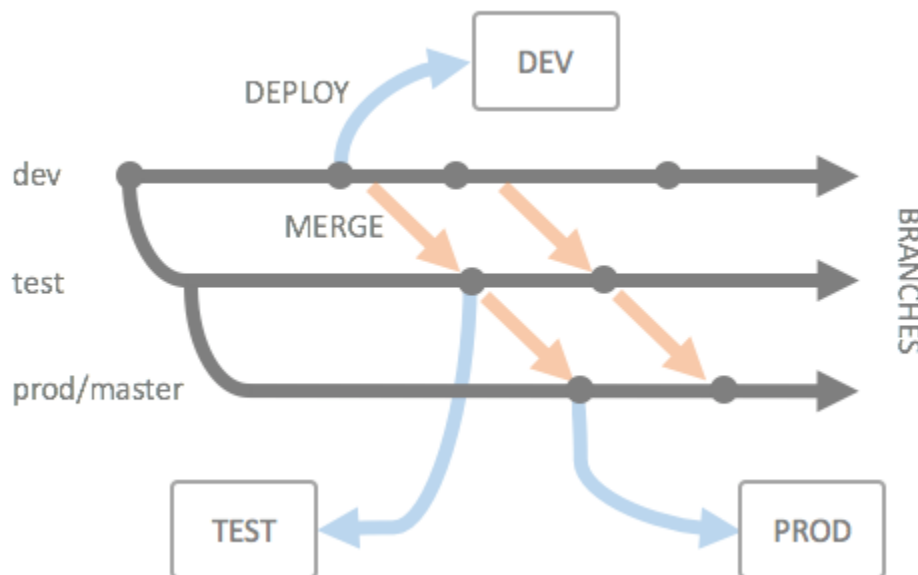
1.1 Les versions

Définition : « le versionnage (équivalent francophone de l'anglais *versioning*) est le mécanisme qui consiste à conserver la version d'une entité logicielle quelconque, de façon à pouvoir la retrouver facilement, même après l'apparition et la mise en place de versions plus récentes. »

Conserver plusieurs versions peut avoir plusieurs utilités. Pour commencer, il est plus facile de suivre les changements qui ont été fait au courant de notre développement. Ensuite, nous pouvons savoir qui a fait quels changements et à quel moment. Finalement, nous pourrions revenir à un stade antérieur de notre logiciel si jamais nous rencontrons des problèmes majeurs lors de la conception d'un module de notre logiciel.

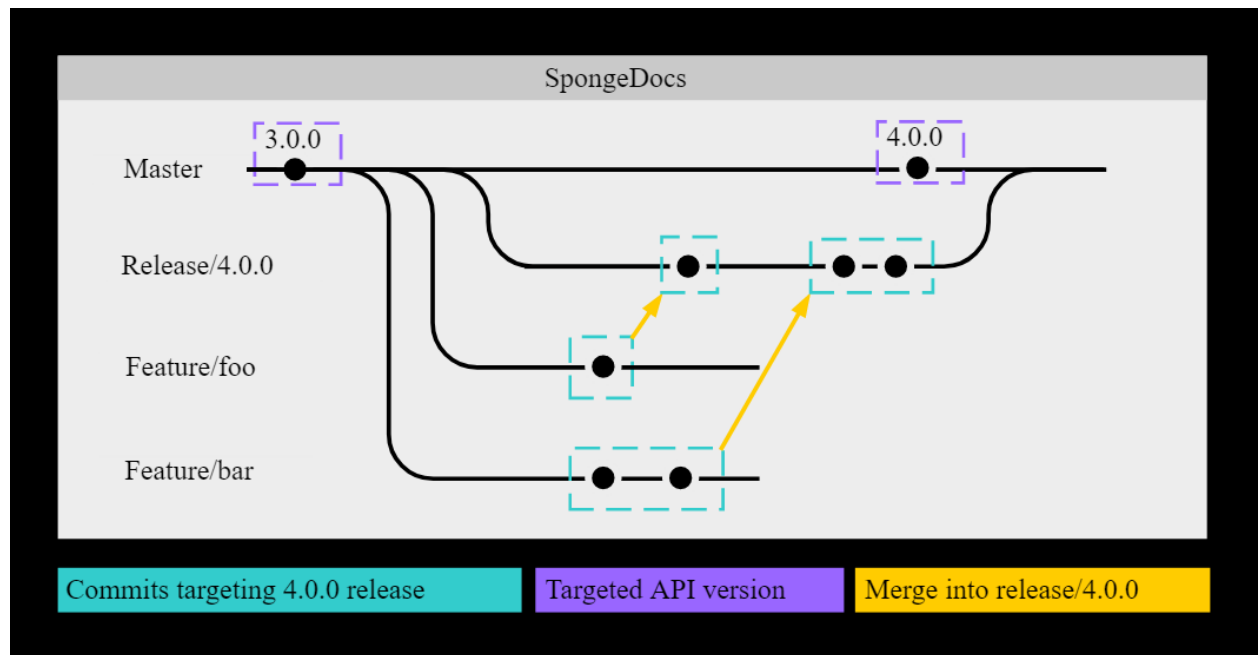
Il faut savoir qu'en entreprise, contrairement à ce que vous avez vécu jusqu'à maintenant, nous avons plusieurs environnements pour le même projet. La raison est simple, nous ne voulons pas que n'importe qui puisse envoyer ses changements sur le serveur de production (l'environnement que les clients utilisent) et introduire des bugs qui pourraient potentiellement coûter très cher.

Il faut savoir que normalement les développeurs travaillent dans ce qu'on appelle l'environnement de développement. Lorsque les changements sont prêts, nous les enverrons dans l'environnement de test et si nos changements passent les tests, ils pourront être envoyés sur l'environnement de production. (Penser aux serveurs de tests des jeux vidéo, une fois que les changements sont prêts nous les pousserons pour tous les joueurs). Voici un exemple (en anglais) :



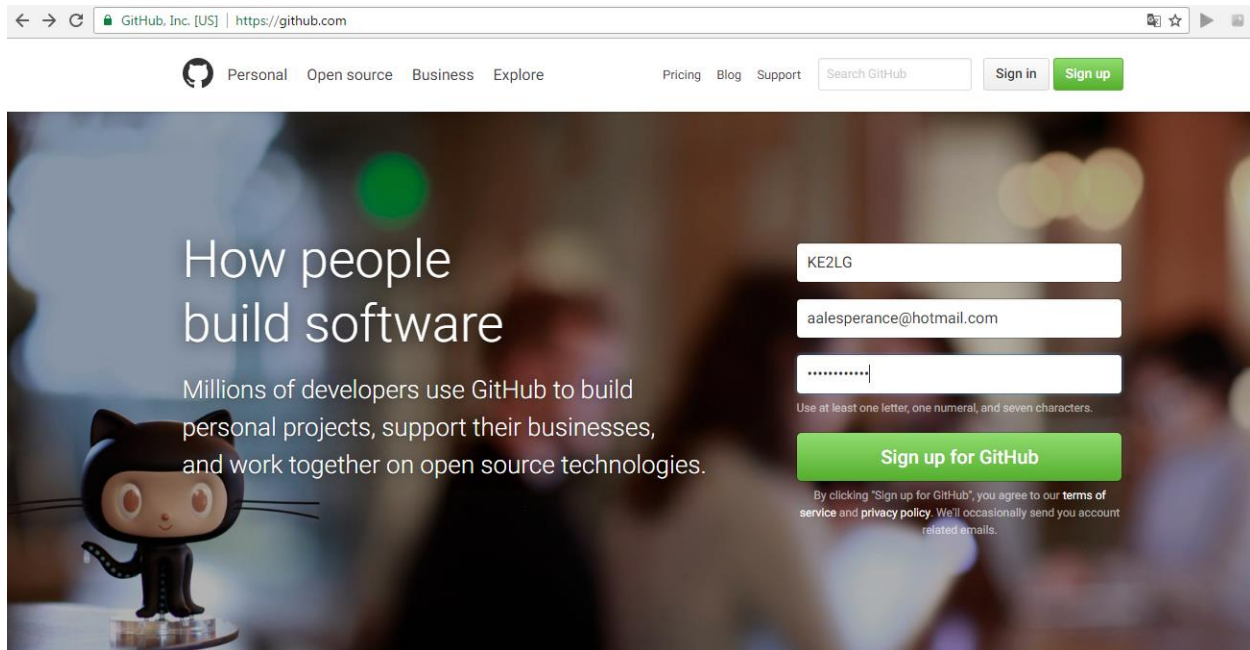
Concentrons-nous seulement sur l'environnement de développement (celui que vous utiliserez le plus dans votre carrière). Dans 99.9999% des cas, nous aurons plusieurs personnes qui utiliseront le même environnement de développement. Si tout le monde travaille sur le même environnement en même temps, si tout le monde travaille sur les mêmes fichiers en même temps, cela pourra causer d'énormes problèmes en plus d'introduire des bugs pour tout le monde.

Pour éviter ce problème, nous pouvons créer ce que nous appelons une branche. Nous prendrons le logiciel dans son état présent et allons créer un environnement de travail « privé » sur lequel nous pourrions travailler sans avoir peur de briser le code pour nos collègues de travail. Une fois notre travail complété, nous pourrions l'intégrer au code qui est en préparation pour la prochaine release.



Parfois, lorsque deux personnes ont travaillé sur le même fichier séparément, un conflit peut se produire, car nous aurons deux versions différentes du même fichier. Il faudra alors fusionner nos documents. La plupart des logiciels de versions auront un système pour faire la fusion de ces documents, mais parfois il faudra le faire manuellement. Lorsque vous travaillez sur vos projets, la communication entre collègues est d'une très grande importance afin d'éviter ces conflits.

1.2 GitHub Desktop (Tutoriel, ancienne version 2018).



The screenshot shows the GitHub homepage with a sign-up form. The form includes three input fields: a username field containing 'KE2LG', an email field containing 'aalesperance@hotmail.com', and a password field with masked characters. Below the password field is a note: 'Use at least one letter, one numeral, and seven characters.' A green 'Sign up for GitHub' button is positioned below the form. At the bottom of the form, there is a small disclaimer: 'By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.'

Personal Open source Business Explore Pricing Blog Support Search GitHub Sign in Sign up

How people build software

Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies.

KE2LG

aalesperance@hotmail.com

.....




Use at least one letter, one numeral, and seven characters.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.

Sur la page principale du site, nous aurons 3 champs : le nom d'utilisateur souhaité, l'adresse courriel utilisé pour le compte, ainsi que le mot de passe désiré.

You've taken your first step into a larger world, @KE2LG.

 Completed Set up a personal account	 Step 2: Choose your plan	 Step 3: Tailor your experience
---	--	--

Choose your personal plan

Unlimited public repositories for free.






Unlimited private repositories for \$7/month. [\(view in CAD\)](#)

Don't worry, you can cancel or upgrade at any time.

Help me set up an organization next
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.
[Learn more about organizations.](#)

[Continue](#)

Both plans include:

-  Collaborative code review
-  Issue tracking
-  Open source community
-  Unlimited public repositories
-  Join any organization

Ensuite, le site posera la question suivante : « Voulez-vous utiliser un compte où tous les répertoires sont publics ou payer pour avoir un compte avec des répertoires privés? ». Ici, nous choisirons un compte gratuit.



GitHub Desktop

[Overview](#) | [Release Notes](#) | [Help](#)

Simple collaboration from your desktop

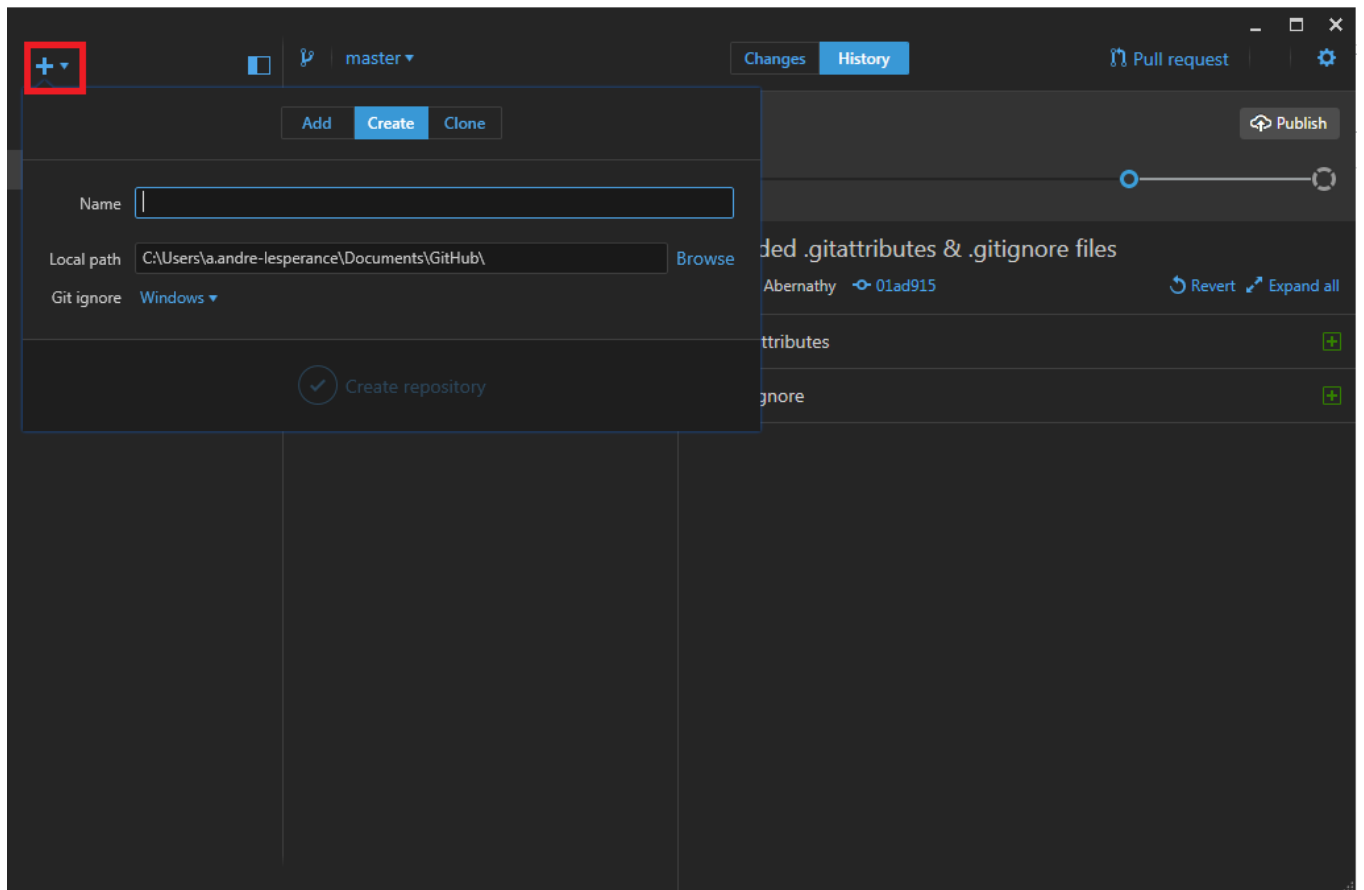
GitHub Desktop is a seamless way to contribute to projects on GitHub and GitHub Enterprise.

Available for Mac and [Windows](#)

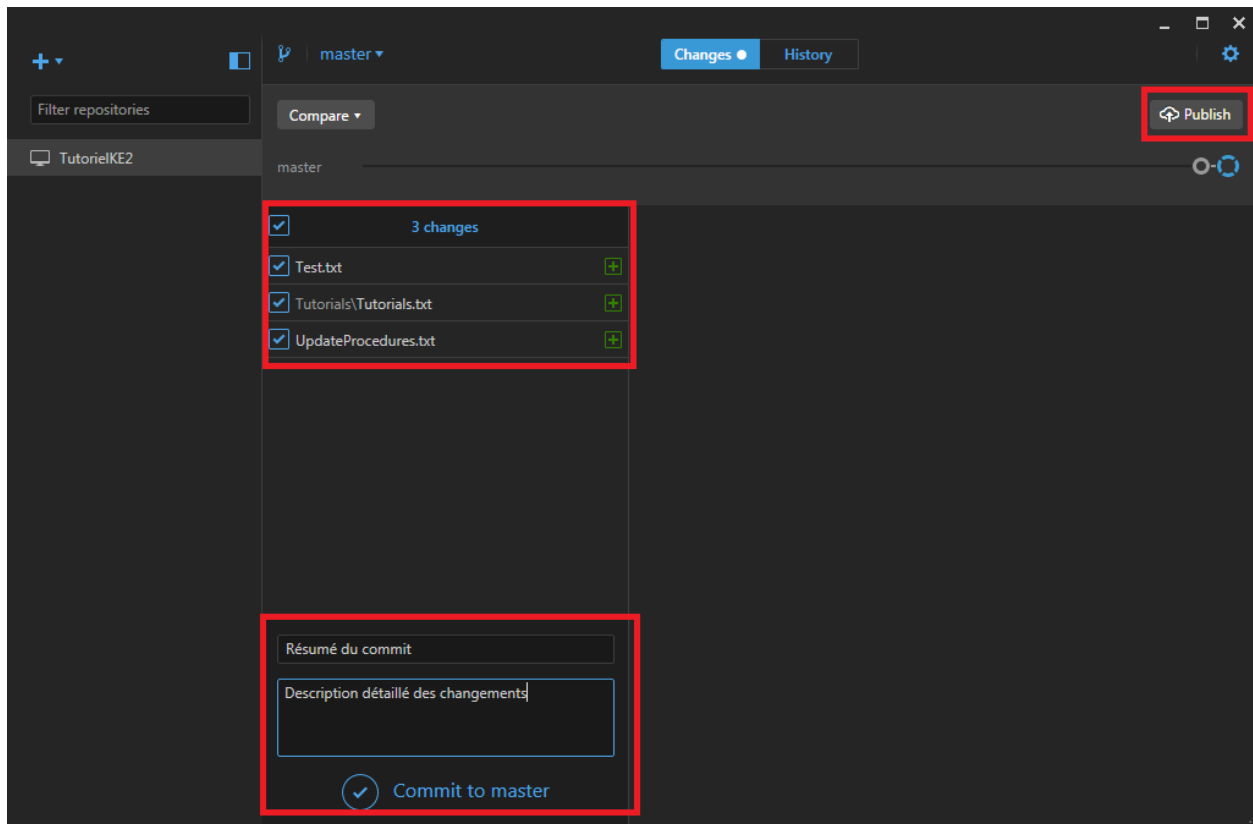
[Download GitHub Desktop](#)
Windows 7 or later

By clicking the Download button you agree to the [End-User License Agreement](#)

Nous téléchargerons ensuite le logiciel GitHub Desktop afin de pouvoir contrôler facilement nos versions à l'aide d'une interface utilisateur facile à utiliser. Vous pouvez le trouver à l'adresse suivante : <https://desktop.github.com/>



Une fois le logiciel téléchargé, GitHub affichera une interface de connexion. Une fois connecté, la page principale s'affichera. Pour créer un nouveau projet, il faut simplement cliquer sur le bouton « + » et choisir le nom du projet et la location de sauvegarde pour notre dossier.



Une fois que notre dossier subira des modifications (ajout, modification ou suppression de fichiers/dossiers), GitHub Office détectera et affichera les changements à l'écran. Nous pouvons donc entrer les informations de nos changements afin de les conserver avec nos modifications et appuyer sur « Commit » pour envoyer notre nouvelle version de manière officielle dans le projet. Ces fichiers seront maintenant accessibles officiellement dans le projet.

Ensuite, il faudra appuyer sur « Publish » afin d'envoyer nos fichiers sur le serveur distant et que nos fichiers soient accessibles par tous les membres du projet. Une fois le projet publié pour la première fois, ce bouton se transformera en « Sync » et téléchargera les changements provenant du dossier distant en même temps d'envoyer les changements pour lesquels on est « commis ».

En cliquant droit sur le titre du projet, dans la section gauche de notre écran, nous pourrions avoir les options d'explorer le dossier sur notre ordinateur dans Explorer ou encore de voir le projet sur GitHub qui nous amènera directement sur le site de GitHub dans le dossier du projet.

1.3 Les branches d'un projet.

Lors de projets de plus grande envergure, nous introduirons le concept de branches à nos outils de version.

Une branche est une solution facile afin d'avoir un environnement de travail sain et sans danger de corrompre le projet principal. Il s'agit d'un clone de notre projet à un certain moment dans lequel nous travaillerons jusqu'à ce que nos modifications aient été prouvées et testées. Une fois que nos modifications ont été testées et sont sans bugs, nous les enverrons à la branche principale (master) de notre projet et nous pourrons continuer à travailler dans notre branche.

Cette technique est utilisée partout en entreprise et nous permettra d'avoir un environnement de production solide et sauf des bugs introduits par inadvertance.

1.4 Les autres outils.

Plusieurs autres outils existent sur le marché. Certains seront aussi Open-Source, distribués et offriront seulement des répertoires publics de base comme GitHub.

D'autres demanderont de payer et de s'intégrer à votre logiciel de développement (Ex. : Team Foundation Server de Microsoft) ou encore seront gratuits, mais fonctionneront tout de même avec vos clients et serveurs afin de garder votre code sur les lieux (ex. : Subversion et ses MULTIPLES clients existant sur le marché, ex. : TortoiseSVN). Dans un futur rapproché, ce sera à vous de choisir l'outil qui conviendra le mieux à vos besoins!

Si vous voulez explorer, voici une liste plus complète des logiciels qui sont disponibles :

https://en.wikipedia.org/wiki/List_of_version_control_software.

Sources

1. https://fr.wikipedia.org/wiki/Version_d'un_logiciel
2. <https://github.com>