
Atelier 1 Démarrer le projet avec GitHub



A un haut niveau, GitHub est un site web et un service de cloud qui aide les développeurs à stocker et à gérer leur code, ainsi qu'à suivre et contrôler les modifications qui lui sont apportées. Pour comprendre exactement ce qu'est GitHub, vous devez connaître deux principes liés :

- Contrôle de version
- Git

Dans cet atelier, nous allons d'abord expliquer ces deux principes. Ensuite, nous en apprendrons plus sur GitHub.

Qu'est-ce que le Contrôle de Version ?

Le contrôle de version aide les développeurs à suivre et à gérer les modifications apportées au code d'un projet logiciel.

Le contrôle de version permet aux développeurs de travailler en toute sécurité à travers les branchements et les fusions.

Avec le branchement, un développeur duplique une partie du code source (appelé le référentiel). Le développeur peut alors apporter des modifications en toute sécurité à cette partie du code sans affecter le reste du projet.

Ensuite, une fois que le développeur a réussi à faire fonctionner correctement sa partie du code, il ou elle peut fusionner ce code avec le code source principal pour le rendre officiel.

Tous ces changements sont ensuite suivis et peuvent être annulés si nécessaire.

Qu'est-ce Que Git ?

Git est un système de contrôle de version open-source spécifique créé par Linus Torvalds en 2005.

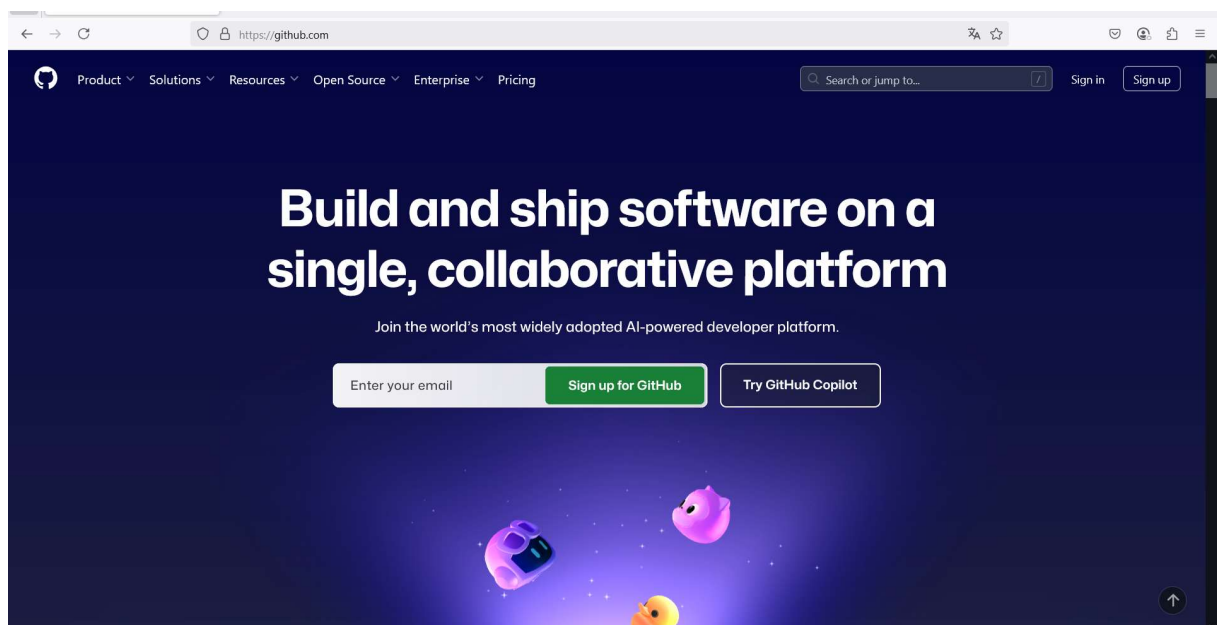
Concrètement, Git est un système de contrôle de version distribué, ce qui signifie que l'ensemble de la base du code et de l'historique est disponible sur l'ordinateur de chaque développeur, ce qui permet des branchements et une fusion facile.

Créez un compte GitHub

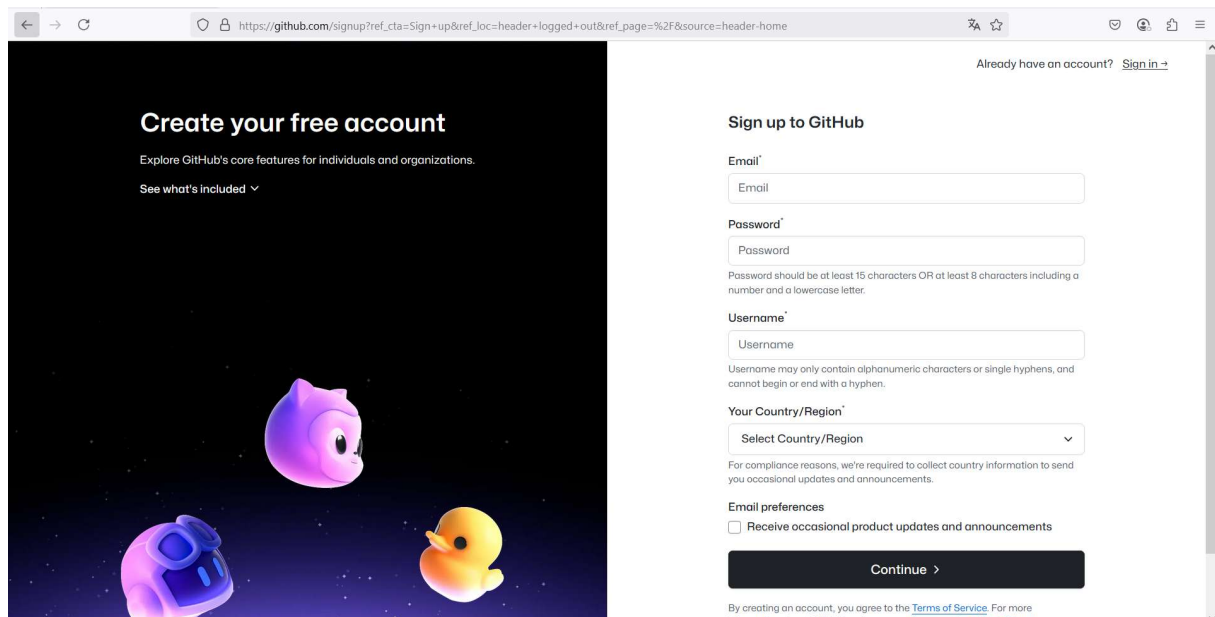
GitHub est un service en ligne qui va vous permettre d'héberger vos dépôts distants.

Pour créer votre compte GitHub, rendez-vous sur la page d'accueil et cliquez sur Sign up.

<https://github.com/>



On vous demandera alors de renseigner un e-mail, un mot de passe et un nom d'utilisateur.



Already have an account? [Sign in](#)

Create your free account

Explore GitHub's core features for individuals and organizations.

[See what's included](#)

Sign up to GitHub

Email*

Password*

Username*

Your Country/Region*

Email preferences

☐ Receive occasional product updates and announcements

[Continue](#)

By creating an account, you agree to the [Terms of Service](#). For more information about GitHub's privacy practices, see the [GitHub Privacy](#) page.

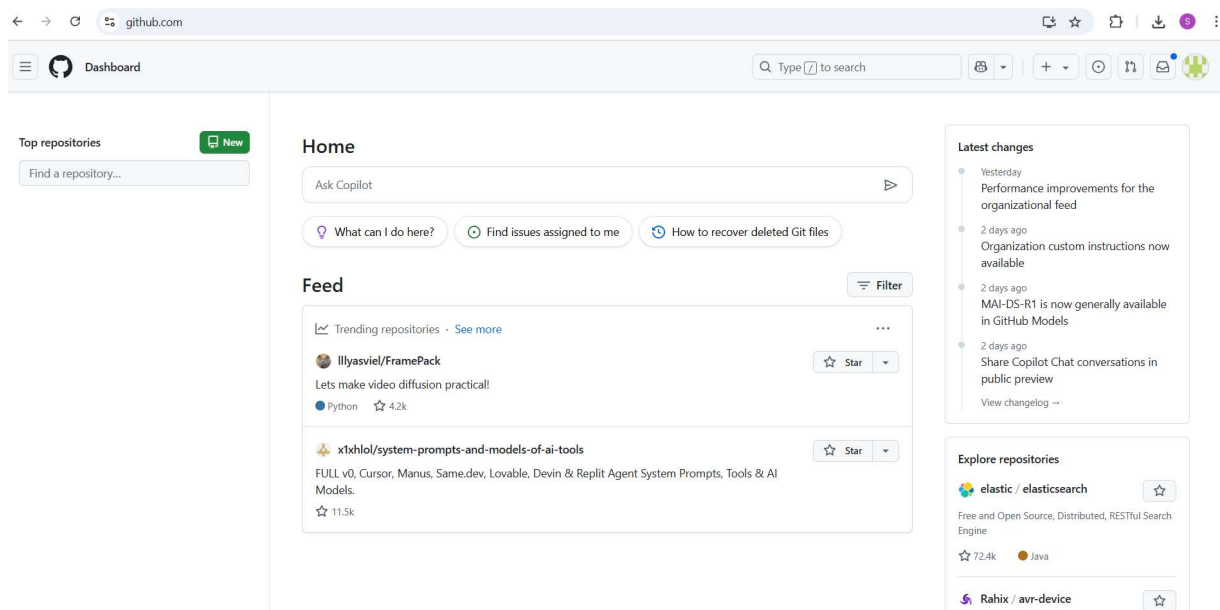
Un code de vérification vous sera envoyé sur votre adresse e-mail afin de confirmer votre identité.

Et voilà, vous êtes à présent inscrit sur GitHub ! Par défaut, GitHub est gratuit. Mais sachez qu'il existe également des offres payantes si vous décidez de passer à la vitesse supérieure.

Votre tableau de bord

Vous pouvez consulter votre tableau de bord personnel pour :

- Suivre les problèmes et extraire les demandes sur lesquelles vous travaillez ou que vous suivez.
- Accéder à vos principaux repositories et pages d'équipe.
- Rester à jour sur les activités récentes des organisations et des repositories auxquels vous êtes abonné.



Dashboard

Search Type [7] to search

Top repositories

Find a repository...

Home

Ask Copilot

What can I do here? Find issues assigned to me How to recover deleted Git files

Feed

Trending repositories · [See more](#)

iliasviel/FramePack

Lets make video diffusion practical!

Python 4.2k

xtxhlo/system-prompts-and-models-of-ai-tools

FULL v0, Cursor, Manus, Samedev, Lovable, Devin & Replit Agent System Prompts, Tools & AI Models.

11.5k

Latest changes

Yesterday

Performance improvements for the organizational feed

2 days ago

Organization custom instructions now available

2 days ago

MAI-DS-R1 is now generally available in GitHub Models

2 days ago

Share Copilot Chat conversations in public preview

[View changelog](#)

Explore repositories

elastic / elasticsearch

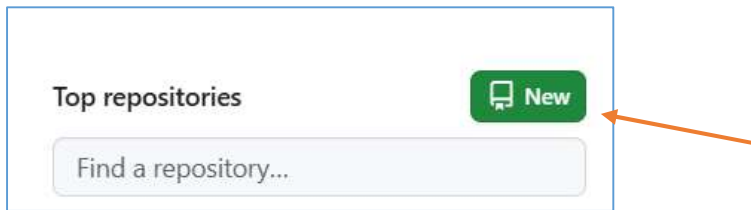
Free and Open Source, Distributed, RESTful Search Engine

72.4k Java

Rahix / avr-device

L'interface repository

L'interface Repositories est l'emplacement où vous pourrez créer et retrouver vos dépôts existants.

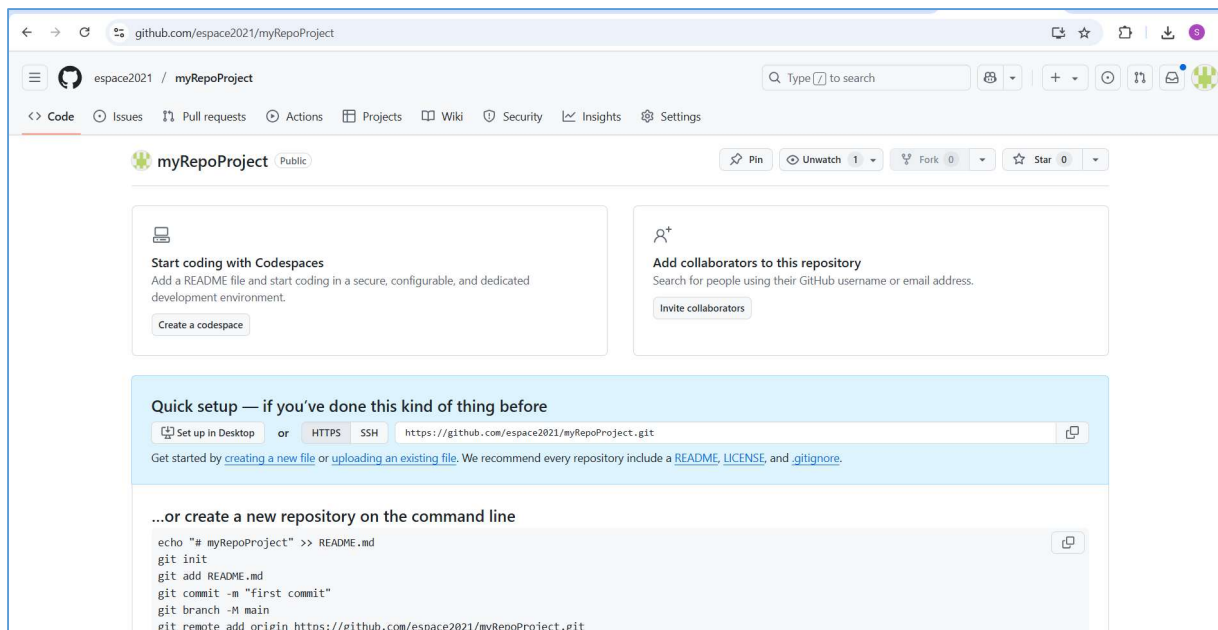
A screenshot of the GitHub 'Create a new repository' form. The form is titled 'Create a new repository' and includes a subtitle: 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)'. Below this, it states 'Required fields are marked with an asterisk (*)'. The form has two main sections: 'Owner' and 'Repository name'. The 'Owner' section has a dropdown menu showing 'espace2021'. The 'Repository name' section has a text input field containing 'myRepoProject', with a green checkmark and the text 'myRepoProject is available.' below it. There is also a link for 'redesigned-fiesta'. Below these fields is a 'Description (optional)' text input field. Further down, there are two radio button options: 'Public' (selected) and 'Private'. The 'Public' option is described as 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is described as 'You choose who can see and commit to this repository.' At the bottom, there is a section 'Initialize this repository with:' with a checkbox for 'Add a README file' and a link 'Learn more about READMEs.'

Dans le champ Repository name, saisir le nom du projet (sans espace et pas de caractères accentués), la description n'est pas obligatoire, il faut ensuite choisir une portée pour le projet :

Public - tout le monde pourra voir le projet

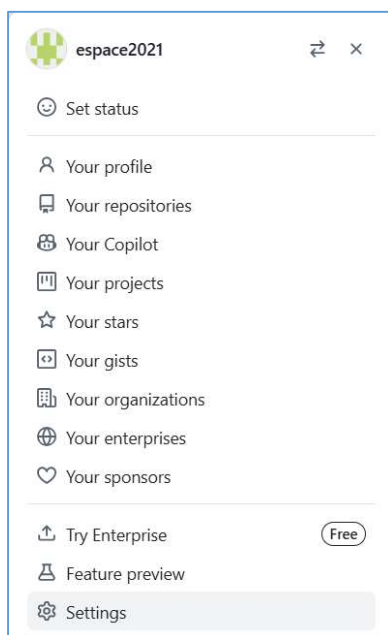
Privé - contrôler qui peut voir le projet en envoyant une invitation à chaque personne

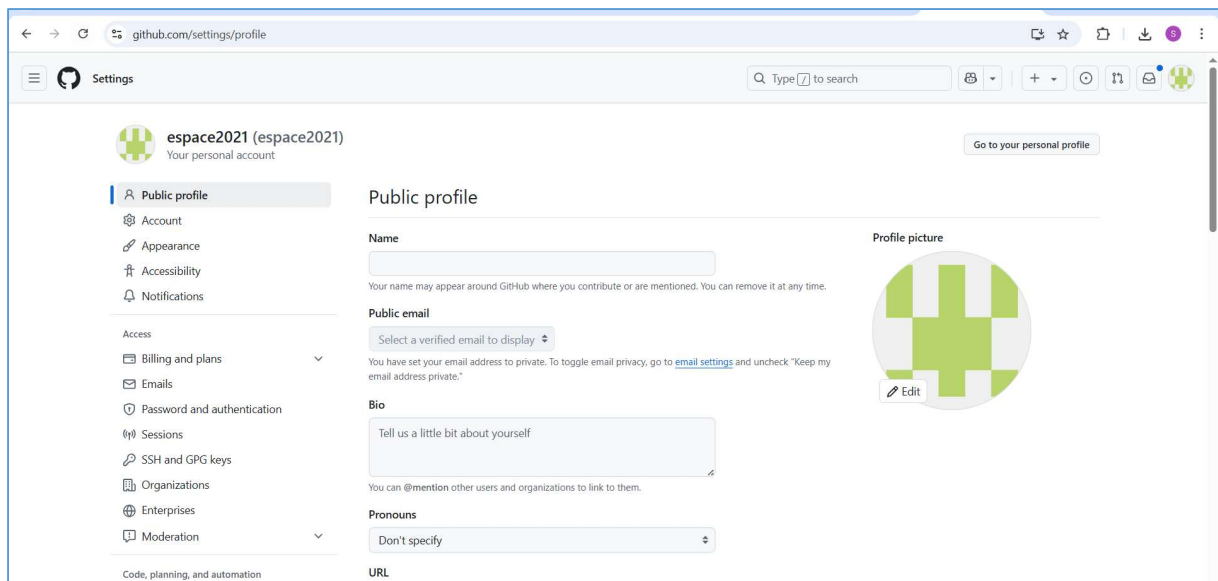
Une fois terminé, on aura une interface comme ici qui te montre les premiers pas pour publier le projet sur ce dépôt :



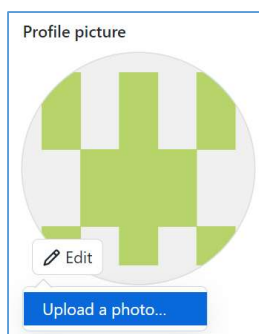
Configuration et paramétrage d'un compte

Accédez aux paramètres de votre compte en utilisant le lien en haut à droite de la fenêtre :

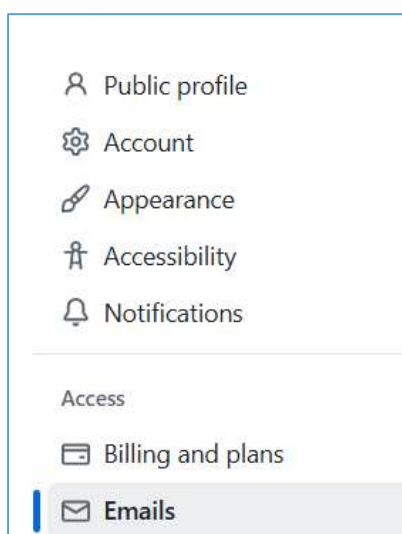


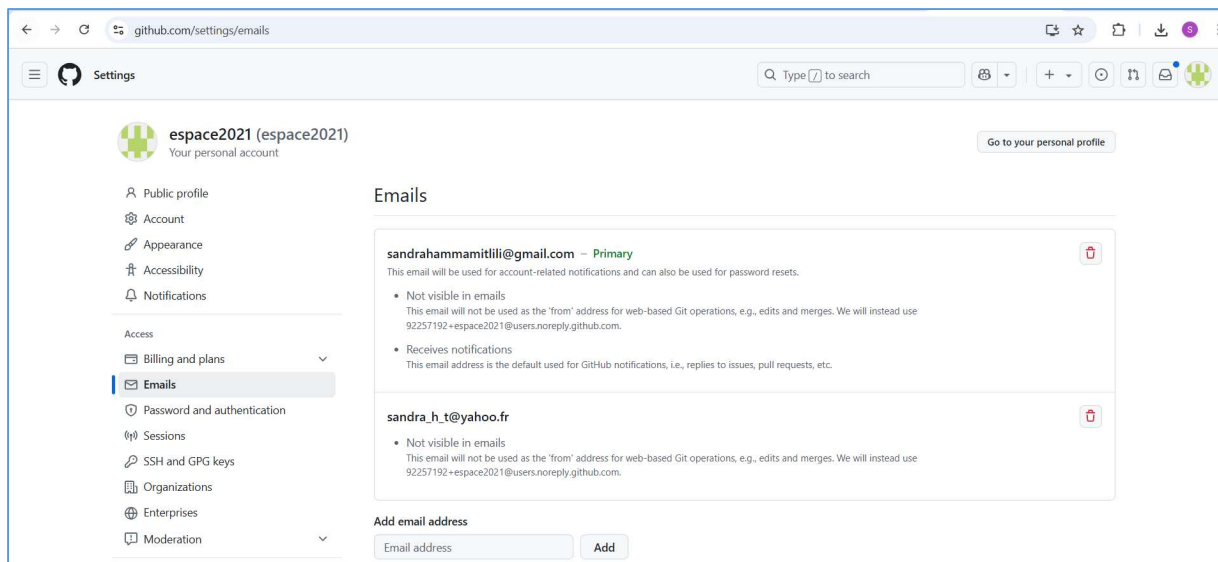


Ensuite, si vous le souhaitez, vous pouvez remplacer l'avatar généré pour vous par une image de votre choix. Sélectionnez la section « Profile » (profil) (au-dessus de la section « SSH Keys ») et cliquez sur « Upload new picture » (télécharger une nouvelle image).



Github utilise les adresses électroniques pour faire correspondre les commits Git aux utilisateurs. Si vous utilisez plusieurs adresses électroniques dans vos commits et que vous souhaitez que GitHub les relie correctement, vous devez ajouter toutes les adresses que vous avez utilisées dans la section « Emails » (adresses électroniques) de la section d'administration.





L'adresse du haut est vérifiée et définie comme adresse principale, c'est-à-dire que ce sera l'adresse utilisée pour vous envoyer toutes les notifications.

La deuxième adresse est non encore vérifiée, ce qui signifie que vous ne pouvez pas en faire votre adresse principale.

La seconde adresse quand elle est vérifiée peut donc aussi être définie comme adresse principale si on l'échange avec la première.

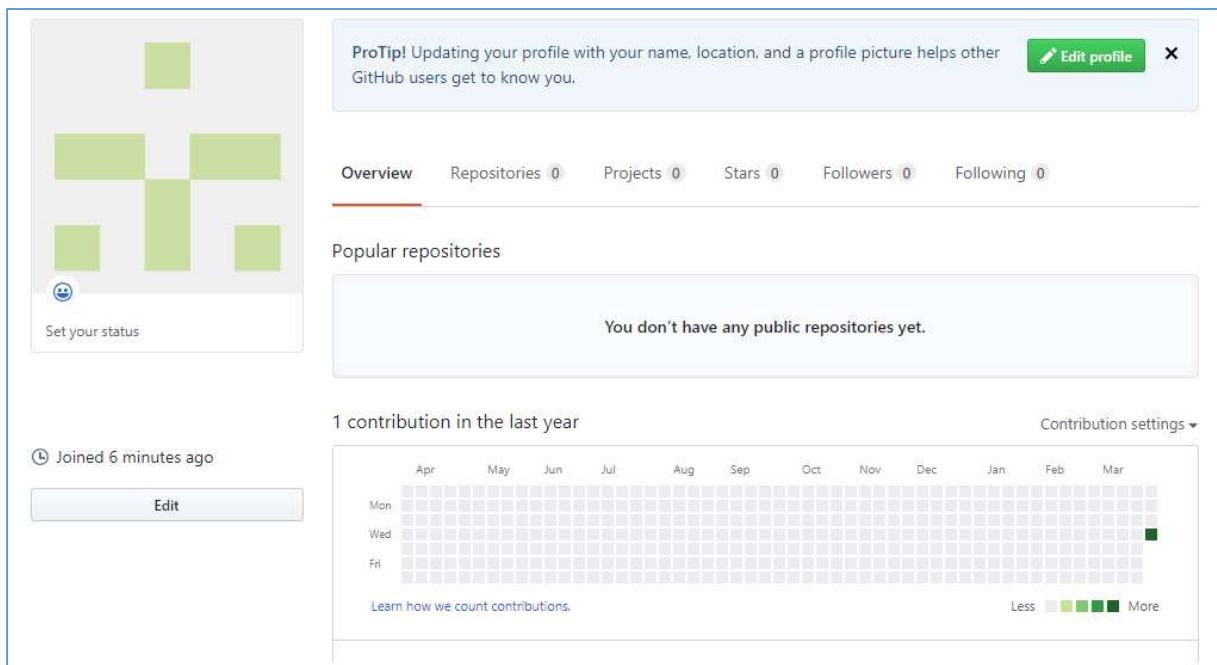
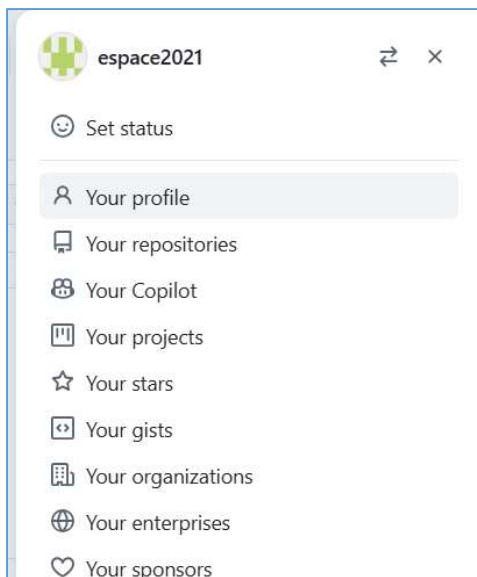
Si GitHub détecte une de ces adresses dans des messages de validation dans n'importe quel dépôt du site, il les reliera à votre compte utilisateur.

Votre profil

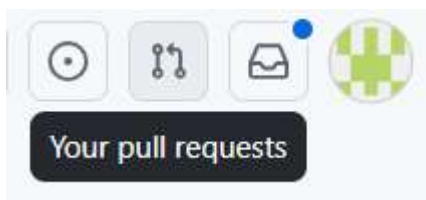
Sur votre profil, vous pourrez éditer vos informations, mais aussi voir le total de vos contributions sur les différents projets.

Les contributions sont toutes les actions sur des repositories que vous allez effectuer. Que ce soient vos repositories, ceux d'autres personnes ou des repositories publics.



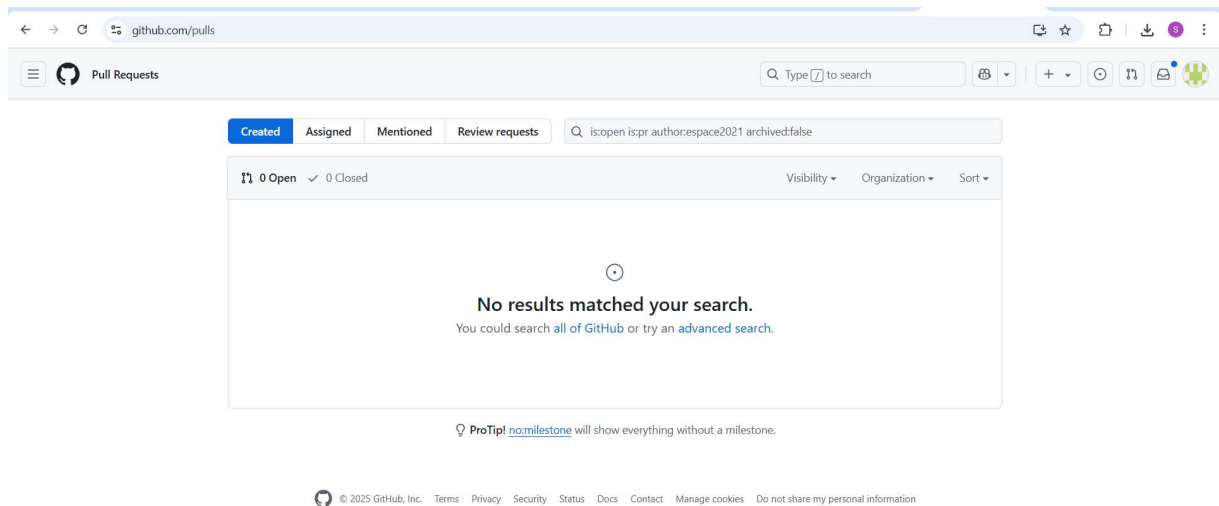


L'onglet Pull requests



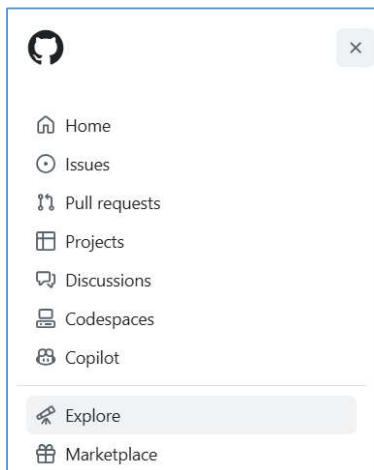
L'onglet Pull requests, quant à lui, permet de faire des demandes de modifications réalisées sur le code.

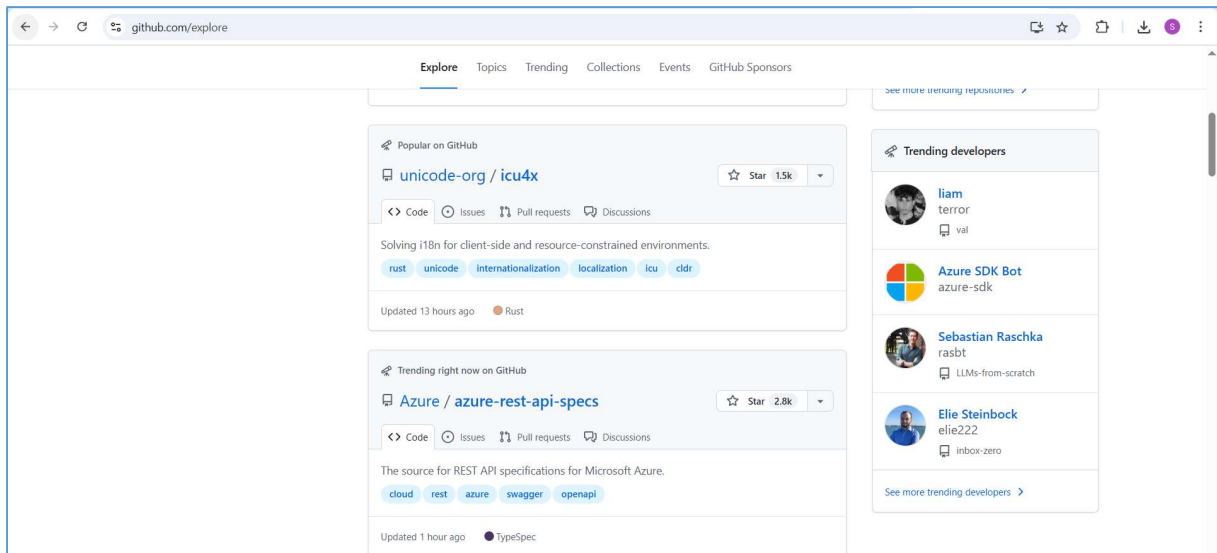
Les pull requests (ou demandes de pull), vous permettent d'informer les autres utilisateurs des modifications que vous avez appliquées à une branche d'un repository sur GitHub, et que vous voulez fusionner avec le code principal.



La fonctionnalité Explore

Via Explore, vous pourrez trouver de nouveaux projets open source sur lesquels travailler, en parcourant les projets recommandés, en vous connectant à la communauté GitHub et en recherchant des repositories par sujet ou par libellé.

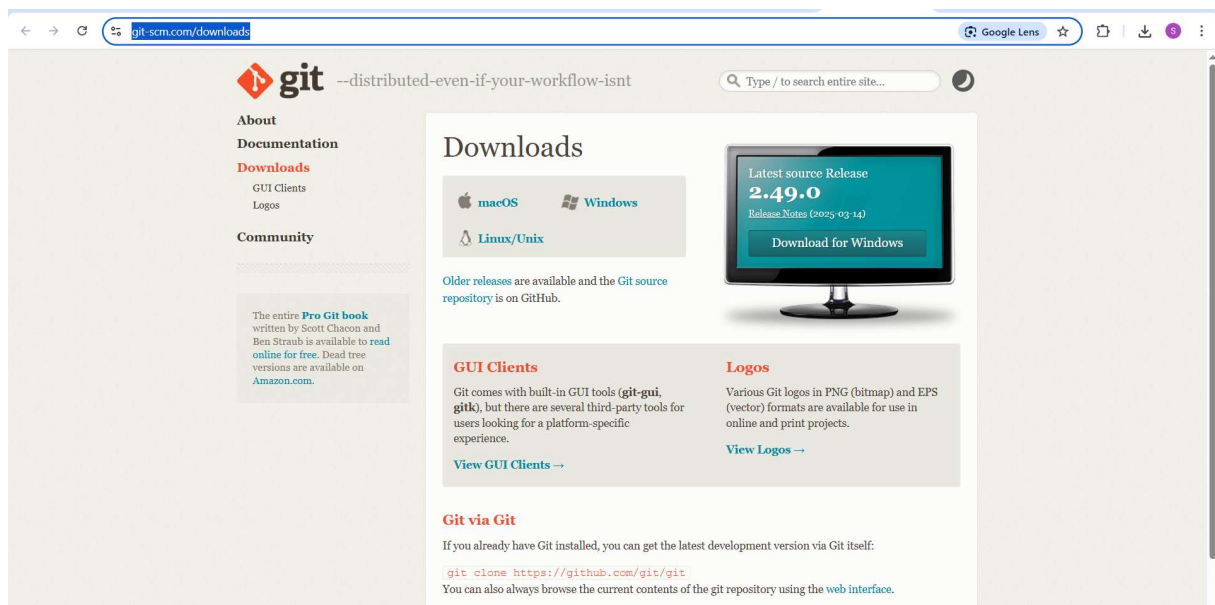




Installer Git sur l'ordinateur

Visiter le lien :

<https://git-scm.com/downloads>

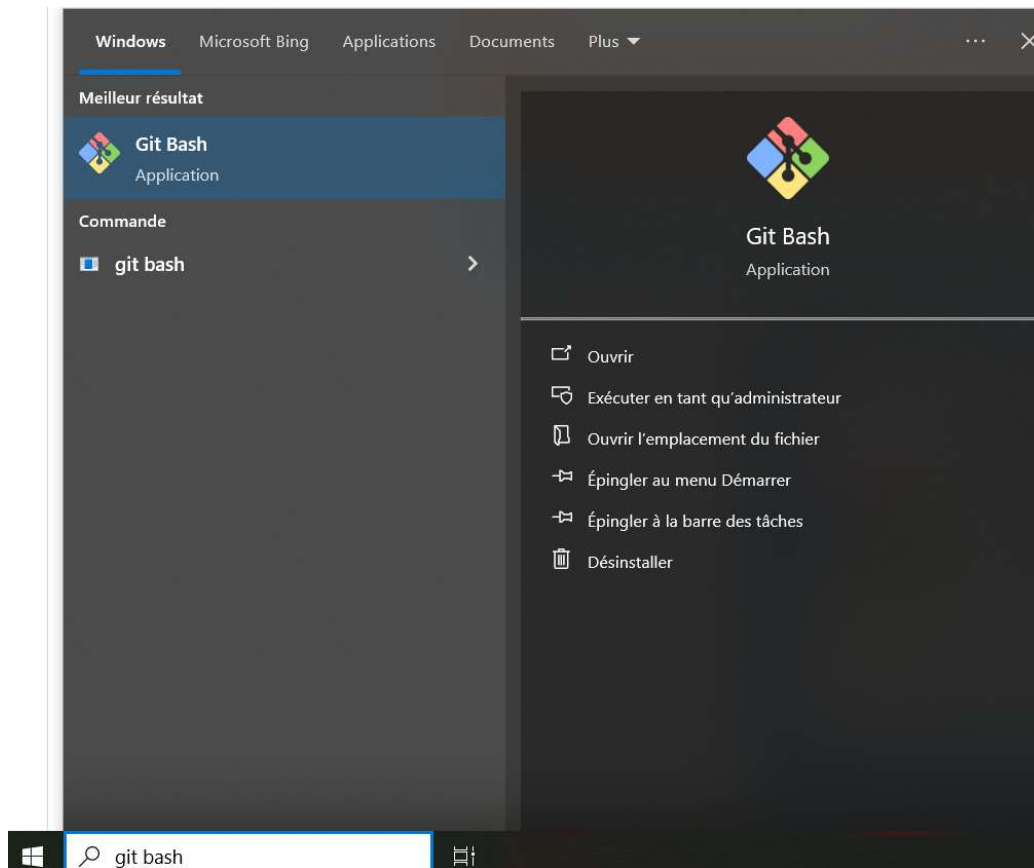


Choisissez et téléchargez la version de Git qui correspond à votre installation : MacOS, Windows ou Linux/Unix.

Exécutez le fichier que vous venez de télécharger.

Appuyez sur Suivant à chaque fenêtre puis sur Installer. Lors de l'installation, laissez toutes les options par défaut, elles conviennent bien.

Git Bash est l'interface permettant d'utiliser Git en ligne de commande.



Ouvrir

Git Bash se lance.



Vérifie l'installation avec :

```
git --version
```

Voici un résumé succinct des commandes Git les plus courantes :

Initialisation et configuration

git init : Initialise un nouveau dépôt Git.

git config : Configure Git (ex : nom d'utilisateur, email).

Statut et historique

git status : Affiche l'état des fichiers (modifiés, en attente...).

git log : Affiche l'historique des commits.

Ajout et validation

git add <fichier> : Ajoute un fichier à l'index (staging).

git add . : Ajoute tous les fichiers modifiés.

git commit -m "message" : Enregistre les modifications avec un message.

Synchronisation

git remote add origin <url> : Ajoute un dépôt distant.

git push origin main : Envoie les commits vers la branche main du dépôt distant.

git pull : Récupère les dernières modifications du dépôt distant.

Branches

git branch : Liste les branches locales.

git branch <nom> : Crée une nouvelle branche.

git checkout <branche> : Bascule vers une branche.

git checkout -b <branche> : Crée et bascule vers une nouvelle branche.

git merge <branche> : Fusionne une branche dans l'actuelle.

Divers

git clone <url> : Clone un dépôt distant en local.

git diff : Affiche les différences entre les fichiers.

git reset : Annule les changements (selon les options).

Ajouter un projet déjà créé

On vous demandera pour la première fois, de s'authentifier à GitHub

1. Naviguer et se mettre à l'intérieur du répertoire du projet dans l'invite de commande (cmd) puis taper :

```
git init
```

pour que git initialise ce dossier en projet git. Cela ajoute les trois arbres nécessaires au suivi de version

- le répertoire de travail (la version actuelle): working directory
- l'index : la liste des choses qui sont préparées (la commande pour ajouter au « stage » est « add »)
- le head qui correspond à ce qui a été « commit » , c'est à dire vraiment envoyé comme version

2. Puis taper :

```
git add .
```

C'est la commande qui « stage » tous les documents, qui les ajoute à l'index. Pour info, on peut sortir de la pile un document avec la commande git reset HEAD votre Fichier

3. Taper :

```
git commit -m "mon premier commit"
```

qui permet de transférer les fichiers en attente dans l'index, vers le Head. Le message permet de savoir à quoi cela correspond.

On utilise l'option -m pour spécifier le message du commit, avec Git, on ne peut pas faire un commit sans spécifier de message. Il faut donc dans ce message expliquer pourquoi on a fait des changements. Ici on utilise juste le message "mon premier commit"

4. Dans l'invite de commande taper :

```
git remote add origin leLienCopiéDuRepository
```

Exemple

```
git remote add origin https://github.com/espace2021/myRepoProject.git
```

va désigner le répertoire distant comme cible du projet.

5. Taper par la suite :

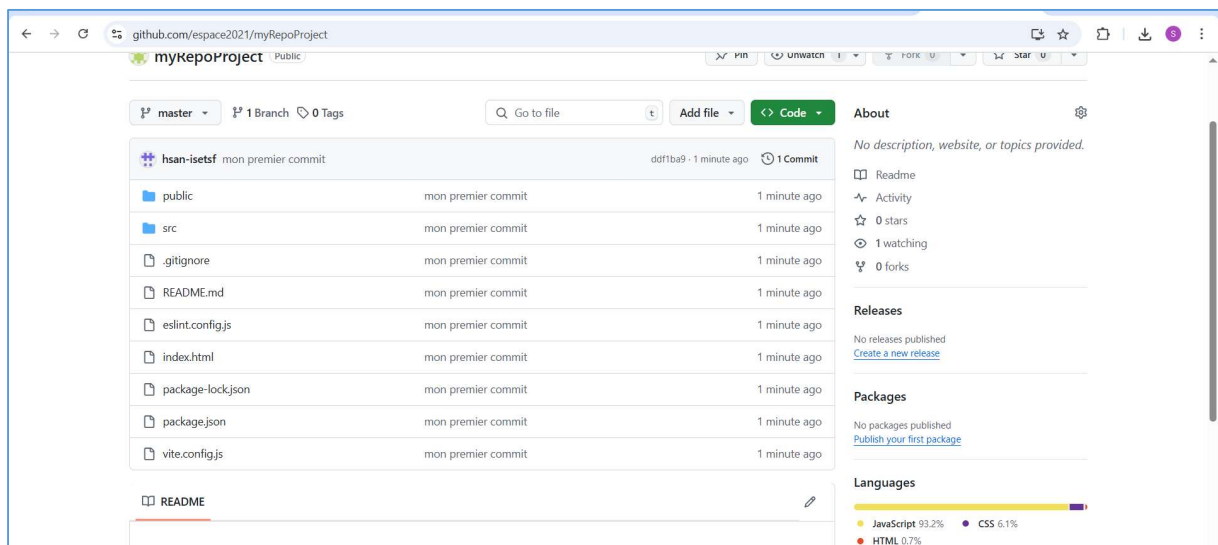
```
git remote -v
```

pour la vérification que les choses fonctionnent dans les deux sens.

6. Enfin taper

```
git push --set-upstream origin main
```

```
Enumerating objects: 47, done.
Counting objects: 100% (47/47), done.
Delta compression using up to 8 threads
Compressing objects: 100% (44/44), done.
Writing objects: 100% (47/47), 549.21 KiB | 12.20 MiB/s, done.
Total 47 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/espace2021/myRepoProject.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```



Pour chaque mise à jour du contenu, pour envoyer les modifications locales apportées au dépôt du serveur :

```
git add .
git commit -m "le nom du commit"
git push
```

Le système de branches

Le principal atout de Git est son système de branches.

Les différentes branches correspondent à des copies de votre code principal à un instant T, où vous pourrez tester toutes vos idées les plus folles sans que cela impacte votre code principal.

Une branche est une “copie” d’un projet sur laquelle on opère des modifications de code.

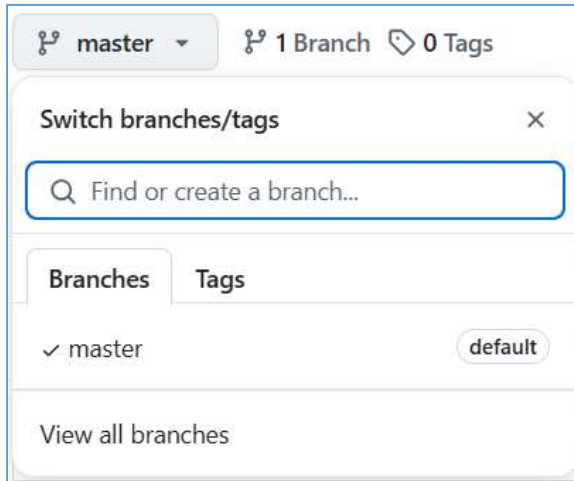
Sous Git, la branche principale est appelée la branche main, ou master.

La branche principale (main ou master) portera l’intégralité des modifications effectuées. Le but n’est donc pas de réaliser les modifications directement sur cette branche, mais de les réaliser sur d’autres branches, et après divers tests, de les intégrer sur la branche principale.

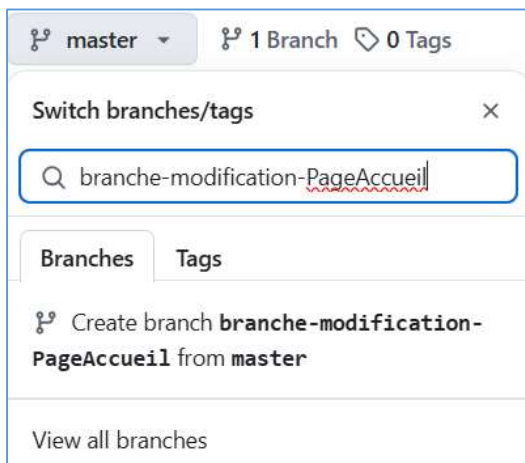
Méthode 1 : Créer une branche depuis l'interface GitHub (web)

Aller sur le dépôt GitHub.

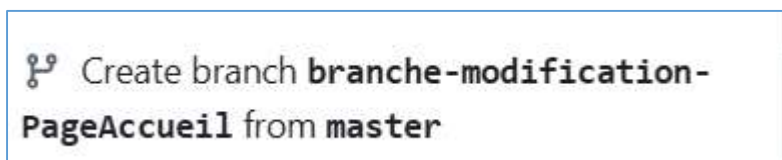
En haut de la liste des fichiers, clique sur le menu déroulant de la branche (souvent nommé main ou master).



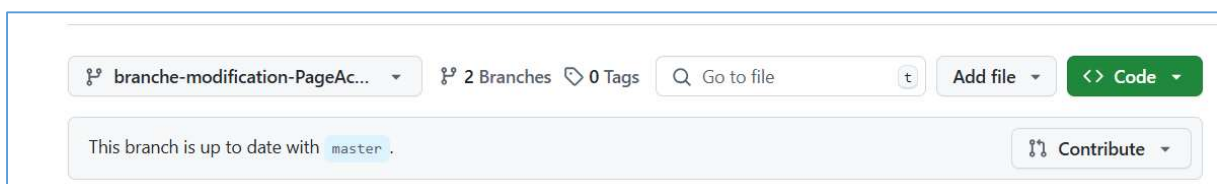
Dans la zone de texte, tape le nom de ta nouvelle branche.



Cliquer sur



La branche est maintenant créée et tu y es automatiquement positionné.



Méthode 2 : Créer une branche en ligne de commande

Créer une branche

```
git checkout -b nom-ta-branche
```

Pousser la branche sur GitHub

```
git push origin nom-branche
```

On peut ensuite faire les modifications, les valider avec git commit, puis faire un push.