Apprenez à utiliser GITHUB

Par Quémard Maël

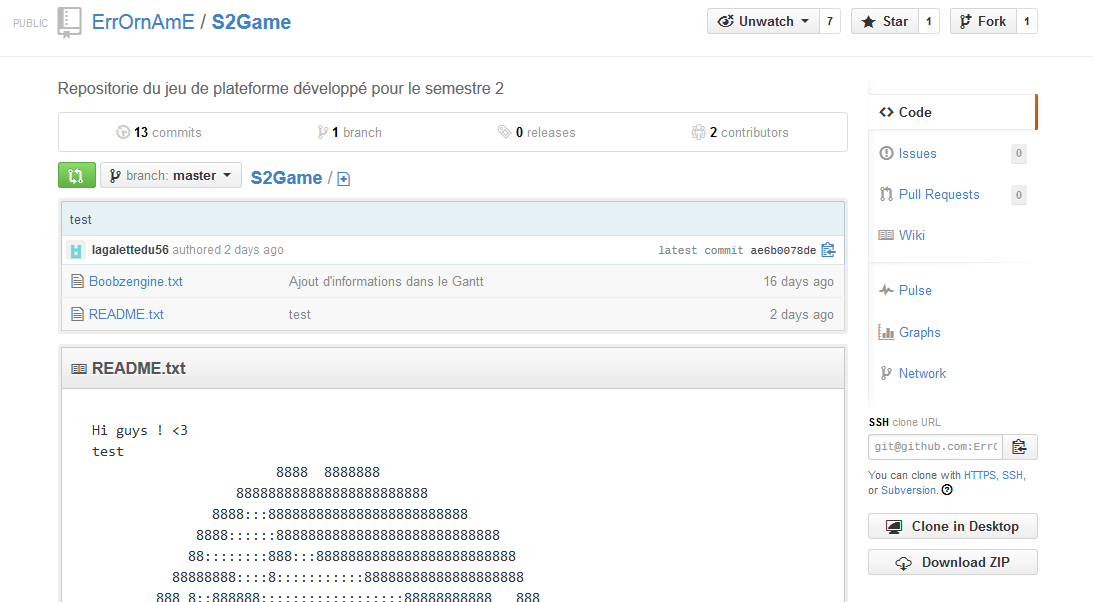
Tout d'abord expliquons ce qu'est Github et a quoi sert-il.

C'est un site et un logiciel qui permet d'héberger des projets versionnés (ce mot n'existe pas dans le dictionnaire mais est utilisé par github et d'autres logiciels de gestion de projet. Cela veux dire faire des versions) et permet de gérer de manière visuelle les collaborateurs et les modifications.

Github peux s'utiliser de plusieurs manières nous allons en voir deux :

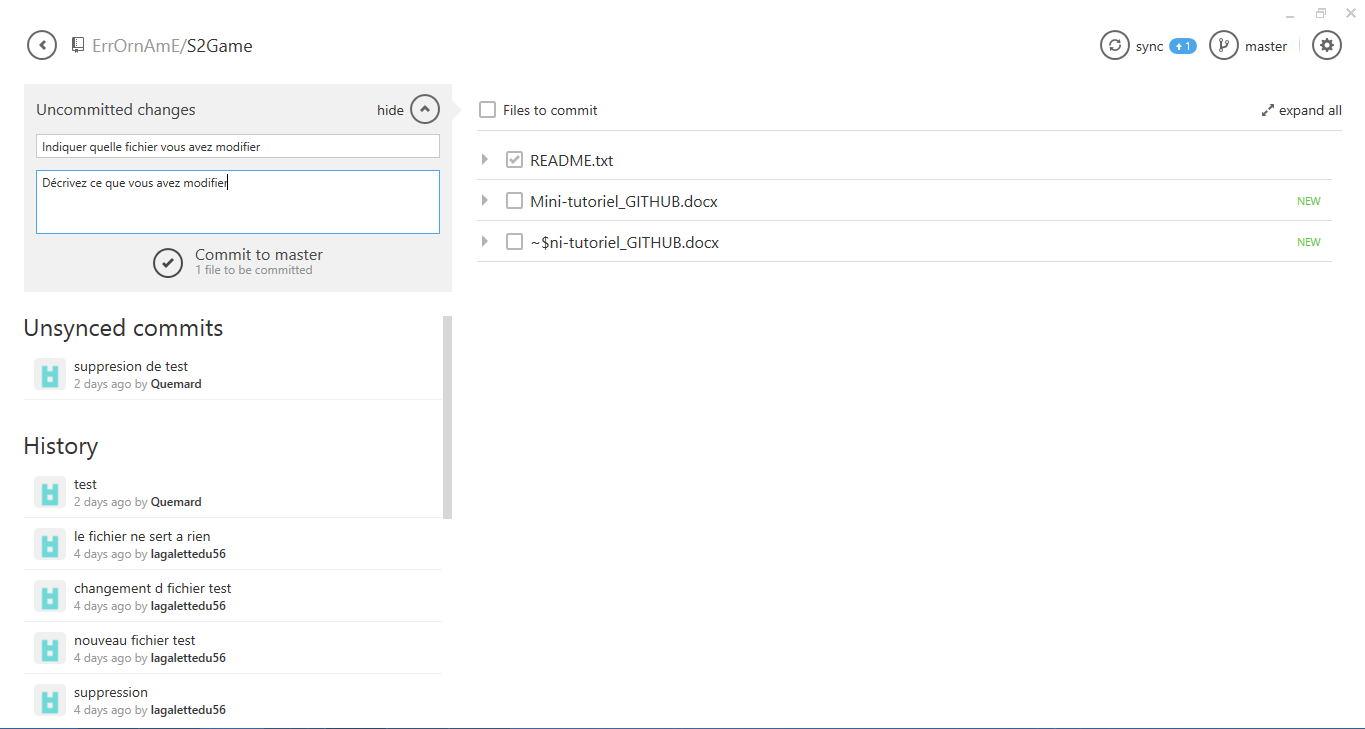
1. **Avec le logiciel**
2. **Avec les lignes de commande.**
3. Première façon de l'utiliser : avec le logiciel

Installer le logiciel puis aller sur le site github et faite un clone du projet sur votre ordi comme ci dessous :



Une fois cela fait, vous avez un dossier avec tout les fichiers du projet sur votre ordinateur ainsi qu'un dossier .ssh avec deux fichiers github\_rsa donc ces fichiers contiennent une clé chacun, une public et l'autre locale, celles ci permettent que lorsque vous démarrez github il n'y ai pas besoin de s'authentifier pour avoir accès a vos projets et a tout leurs fichiers.

Maintenant il ne vous reste plus qu'a bosser sur le projet tranquillement sur votre ordinateur et une fois que vous voulez mettre sur le serveur public il vous suffit de faire un commit (soumission) comme ci-dessous.

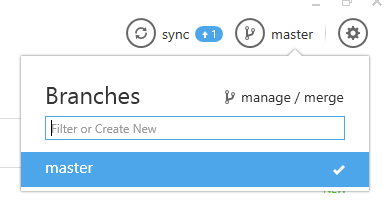


Ici vous synchroniser votre répertoire github local avec le répertoire public

Voici ce qu'il faut faire pour soumettre vos modifications

Voici les fichiers que vous avez modifier et qui ne sont pas mis sur le serveur

Maintenant nous allons voir comment faire un fork (soit en traduction littéral "fourche" dans notre cas nous allons plutôt appeler ça "branche"), donc un fork c'est faire a partir d'un objet (ce qui peut être un projet) de même base une dérivé de celui ci et cela vous permet de faire vos modifications sur le projet et de pouvoir les mettre sur le serveur public sans toucher a la branche principale (master) du projet. Mais une dérivé peut être incluse dans la branche principale du projet.



Ici vous pouvez filtrer les branches qu'il y a dans le projet ainsi qu'en créer de nouvelle juste en lui donnant un nom.

Une fois que vous avez créer votre nouvelle branche il vous faut faire un commit.

1. Deuxième façon de l'utiliser : avec les commandes.

Il vous faut quand même avoir installer le logiciel avec le Shell pour pouvoir utiliser les commandes.

Pour commencer voila mes sources pour les commandes : gitref.org

Il vous faut tout d'abord faire :

**$ git init**

cela va permettre d'initialiser votre répertoire git.

Pour faire un clone du projet sur votre ordinateur :

**$ git clone [url]**

exemple :

**$ git clone git://github.com/ErrOrnAmE/S2Game.git**

Pour suivre des fichiers commencer en faisant

**$ git add \***

Cela va vous permettre de suivre tout les fichiers du projet et que vos modifications soit prise en compte par votre ordinateur et votre répertoire.

Pour soumettre (commit) vos modifications il vous faut d'abord faire un config si ce n'est pas déjà fait.

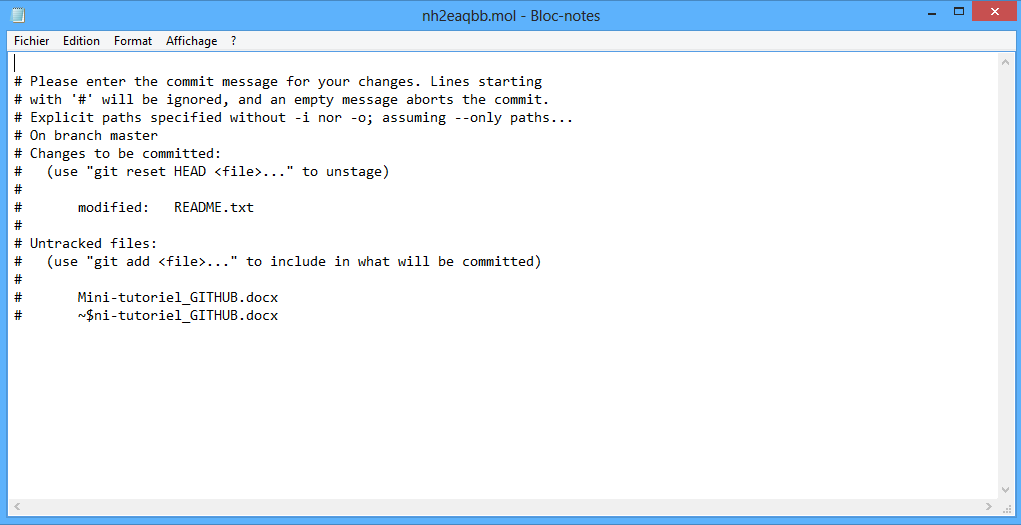
**$ git config --global user.name 'votrenom'**

**$ git config --global user.email you@email.com**

Ici vous avez donnez votre nom et votre adresse mail a git pour vous identifier lors de vos modifications.

**$ git commit [nom du fichier]**

avec l'option **-m** vous pourrez mettre directement votre message de ce que vous avez changés entre simple quote. Sinon il va vous apparaitre une fenêtre tel que celle ci-dessous :



**Ici entrer le nom du fichier modifié sur la première ligne puis a la troisième ligne décrivez vos modifications.**

Puis vous pouvez sauter l'étape du add en utilisant simplement l'option **-a**.

Pour connaitre le statu complet de votre répertoire :

**$ git status**

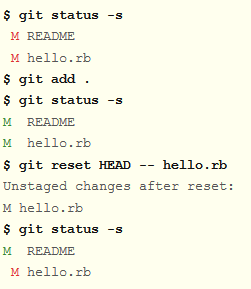
vous pouvez ajoutez l'option **-s** pour voir simplement les fichiers modifier et dont elles n e sont pas sur le serveur

Pour connaitre les différences qu'il y a entre les fichiers du dépôt (serveur) et celui qui est dans votre répertoire :

**$ git diff**

Avec l'option **--cached** vous pourrez voir aussi celle qui avez déjà mis en suivi. Ou encore l'option **--stat** pour voir le nombre d'insertion et de suppression effectué par vos soin.

Si vous faite une erreur comme faire un commit de deux fichiers alors que vous vouliez le faire en deux étapes vous pouvez l'annuler comme par exemple avec ceux-ci :



Pour les branches voici quelques commandes :

**$ git branch**

cela vous permet de voir les différentes branches qu'il y a dans le projet.

**$** **git branch mybranch**

cela vous permet de créer une nouvelle branche ici appeler "mybranch".

**$ git checkout mybranch**

cela vous permet de mettre la branche "mybranch" active.

**$ git checkout -b [nom de la branche]**

cela vous permet de créer et en même temps de la mettre active.

**$ git checkout [nom de la branche]**

cela permet de changer de branche active.

**$ git merge [nom de la branche]**

cela permet de fusionner la branche avec la branche par défaut (master)

**$ git branch -d [nom de la branche]**

cela permet de supprimer une branche.

**$ git branch -v**

Cela permet de voir les derniers commit sur les branches.

Vous pouvez supprimer, déplacer.. les fichiers dans git en utilisant les commandes normal et en rajoutant git avant. Par exemple :

**$ git rm readme.txt**

Voilà je vous ai donner les commandes que nous allons principalement utiliser, si jamais il y aurai besoins de choses que je n'ai pas mis dans ce tuto vous pouvez allez voir sur gitref.org ou encore sur pro git.