

Formation Git

VIII.a - Github



Arnaud MERCIER
arnaud.mercier@hexotech.fr



Nous allons découvrir dans ce chapitre la plateforme github qui permet d'héberger des dépôts git et qui apporte des outils complémentaires autour de git pour la gestion de votre projet comme par exemple:

- gestion de tickets
- CI/CD
- Documentation
- Gestion des droits

Historique de la plateforme



<https://github.com>

2008

Création

Création de la plateforme
d'hébergement en ligne de dépôts
Git

2011

1 Million

La plateforme compte plus de 1
million de projets. Presque
essentiellement OpenSource

2013

10 Millions

La plateforme compte plus de 10
millions de projets. De plus en
plus de projets propriétaires

2018

Rachat par Microsoft

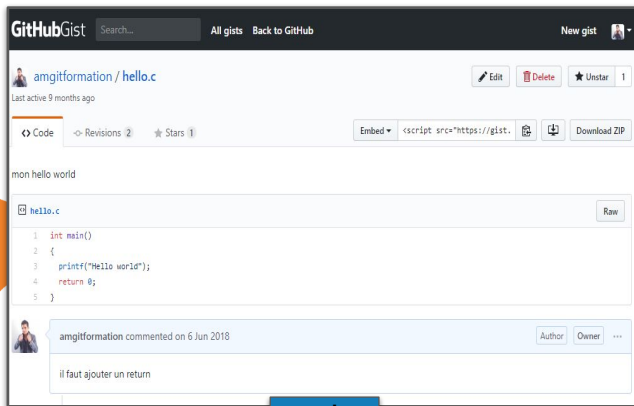
Rachat de la plateforme par
Microsoft pour 7,5 milliards de
dollars

Github est un service en ligne qui permet entre autre d'héberger des dépôts Git. Il est totalement gratuit pour des projets ouverts au public mais il propose également des formules payantes pour les projets que l'on souhaite rendre privés.

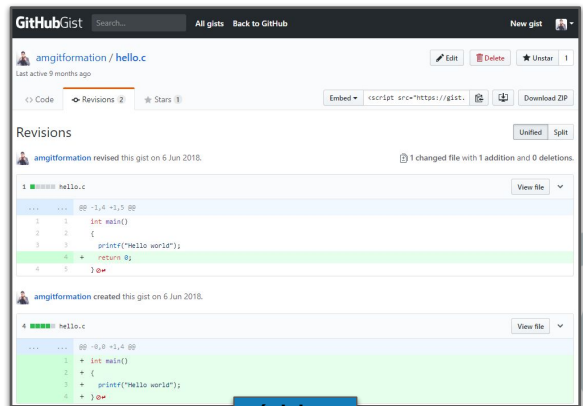
Github propose également de nombreux autres services très intéressants comme par exemple:

- Partager du code source avec d'autres développeurs.
- Signaler et gérer les problèmes ou bugs de votre code source via les issues.
- Partager des portions de code via les Gists
- Proposer des évolutions pour un projet open source.
- Et bien plus encore

Les gists



code

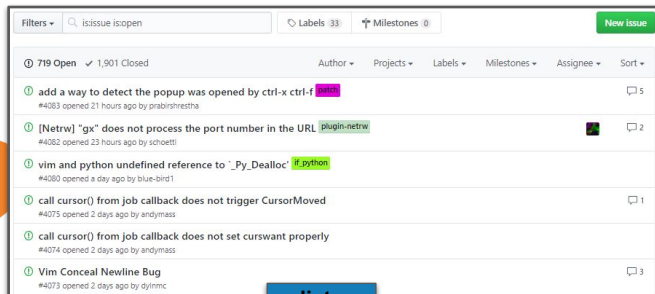


révisions

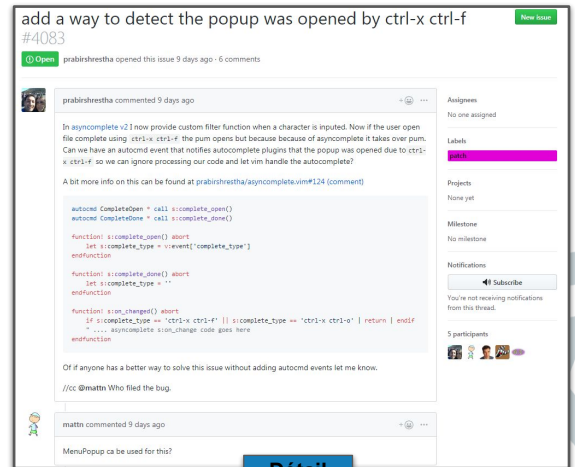
l'outil gists vous permet de partager une portion de code. par exemple pour un forum d'entraide ou encore pour un site de formation. Il permet en plus d'échanger à l'aide d'un système de commentaire associé à chaque gists. Pour finir, il permet en plus de gérer de faire de la micro gestion de conf.

Attention: les gists correspondent à des portions de code et non des projets complets

Les issues



liste



Détail

GitHub permet également d'associer aux dépôts un gestionnaire d'anomalie. Celui-ci permet alors aux consommateurs de votre dépôt de remonter des bug ou des demandes d'évolution.

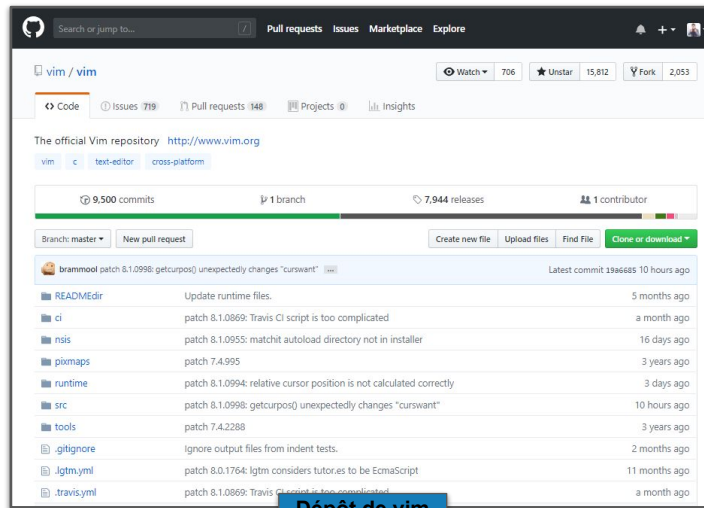
Il permet également d'échanger autour de chaque issue à l'aide d'un système de commentaire.

On retrouve également d'autre aspects génériques de la gestion d'anomalie comme par exemple:

- criticité
- reproductivité
- assigner
- état
- ...

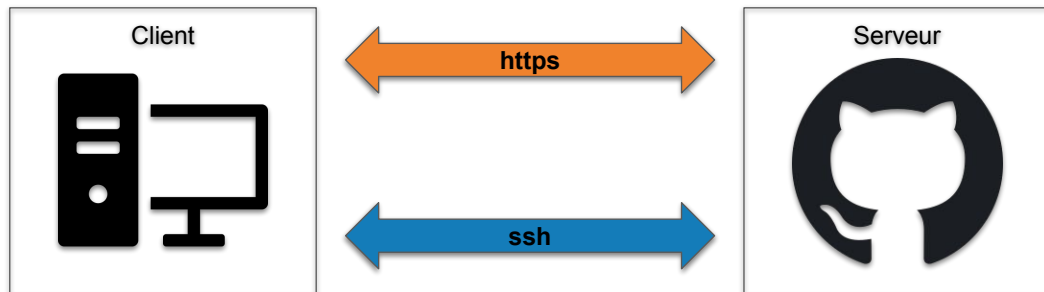
Pour finir, il est possible d'indiquer le numéro de l'issue dans le message d'un commit sous la forme "#id message". Cela permet de faire le lien entre le commit et l'issue dans l'outil.

Les dépôts



le principal intérêt de Github est de pouvoir héberger des dépôts Git. Une interface web permet de très simplement créer et gérer des dépôts.

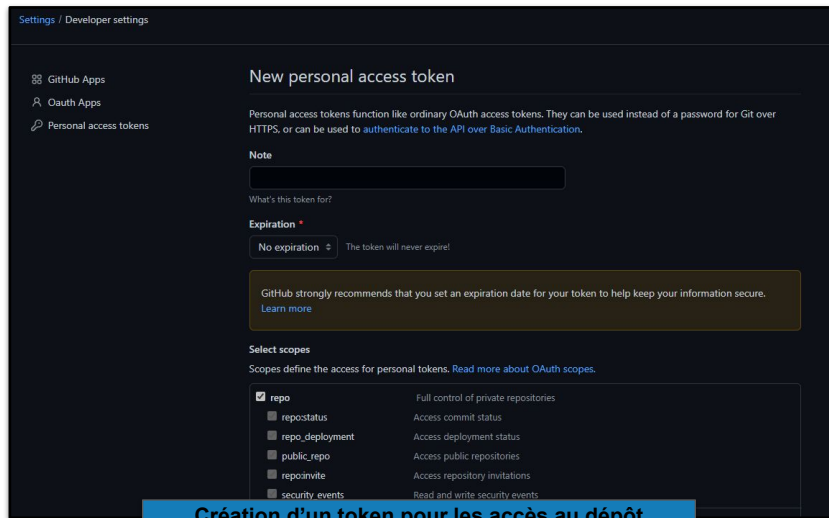
Communication avec Github



Pour travailler avec les dépôts git, il existe deux modes de communication possible:

- https (login + token)
- ssh (clée publique et privée)

https et token



The screenshot shows the 'New personal access token' page in the GitHub Developer Settings. The page has a dark theme. On the left, there's a sidebar with 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens'. The main content area is titled 'New personal access token'. It includes a 'Note' section, a 'What's this token for?' field, an 'Expiration' dropdown set to 'No expiration', and a warning box stating 'GitHub strongly recommends that you set an expiration date for your token to help keep your information secure.' Below this is the 'Select scopes' section, which lists various permissions with checkboxes. The 'repo' scope is checked, and its description 'Full control of private repositories' is shown. Other scopes include 'repo_status', 'repo_deployment', 'public_repo', 'repo_invite', and 'security_events'.

Settings / Developer settings

GitHub Apps
OAuth Apps
Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note

What's this token for?

Expiration *

No expiration - The token will never expire.

GitHub strongly recommends that you set an expiration date for your token to help keep your information secure. [Learn more](#)

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo_status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo_invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events

Création d'un token pour les accès au dépôt

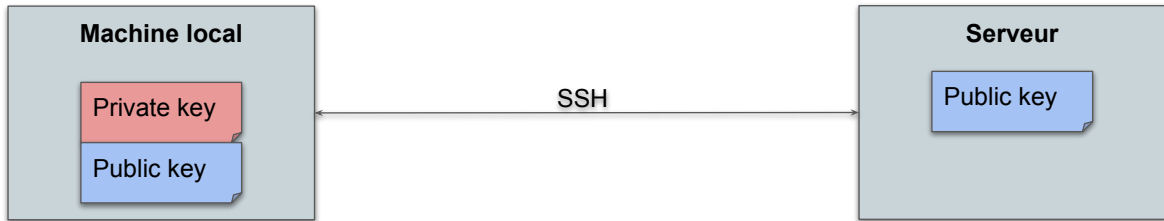
<https://github.com/settings/tokens>

Pour la connexion en https, il faut s'identifier via son login github et via un token personnel en guise de mdp.

Pour créer un token, il faut se rendre dans les settings de votre settings->developper settings->access tokens

Il faut alors donner un nom et sélectionner les 'scopes'. Pour travailler avec un dépôt git, cochez "repo"

SSH



ssh-keygen

```
Generating public/private rsa key pair.  
Enter file in which to save the key (/Users/schacon/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /Users/schacon/.ssh/id_rsa.  
Your public key has been saved in /Users/schacon/.ssh/id_rsa.pub.
```



les clés sont dans le dossier <home>/ssh

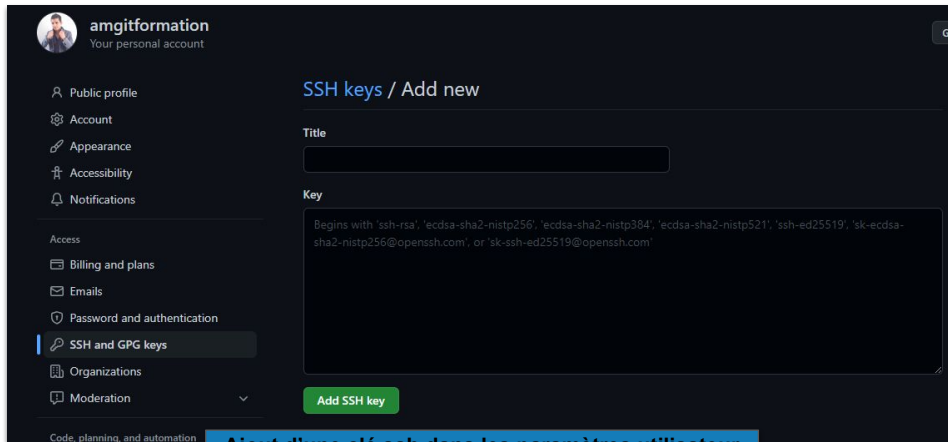
Le système ssh repose sur deux clés, une publique et une privée. La privée reste chez le générateur et la publique est envoyée aux autres systèmes souhaitant communiquer de manière sécurisée avec le générateur de la clé ssh.

Les messages sont alors cryptés avec une des deux parties de la clé (publique ou privée) et seule l'autre partie de la clé peut décoder le message. Cette méthode est très robuste et évite d'avoir un système de login/mot de passe.

Pour générer la clé sur son pc, il faut utiliser la commande "ssh-keygen". Une fois la génération terminée, copier la partie publique et la coller dans l'interface web de Gerrit dans l'onglet settings->ssh.

Conseil: si vous avez plusieurs PCs, inutile de générer plusieurs fois des clés ssh, utilisez la même clé privée sur vos différents PCs.

SSH



Ajout d'une clé ssh dans les paramètres utilisateur

<https://github.com/settings/keys>

Une fois la génération terminée, copier la partie publique et la coller dans l'interface web de github dans l'onglet user settings -> ssh keys

Conseil: si vous avez plusieurs PCs, inutile de générer plusieurs fois des clés ssh, utilisez la même clé privée sur vos différents PCs.

Alternatives à Github



<https://bitbucket.org/product>

Fonctions:

Dépôt Git
Gestion d'anomalie
Relecture
Intégration continue
...



<https://gitlab.com/gitlab-org/gitlab-ce/>

Fonctions:

Dépôt Git
Gestion d'anomalie
Relecture
Intégration continue
...

Plusieurs alternatives existent pour Github comme par exemple Bitbucket et GitLab

TP 7.1

Scénario

Vous souhaitez contribuer à la communauté OpenSource via GitHub sur le dépôt:

https://github.com/amgitformation/formation_git

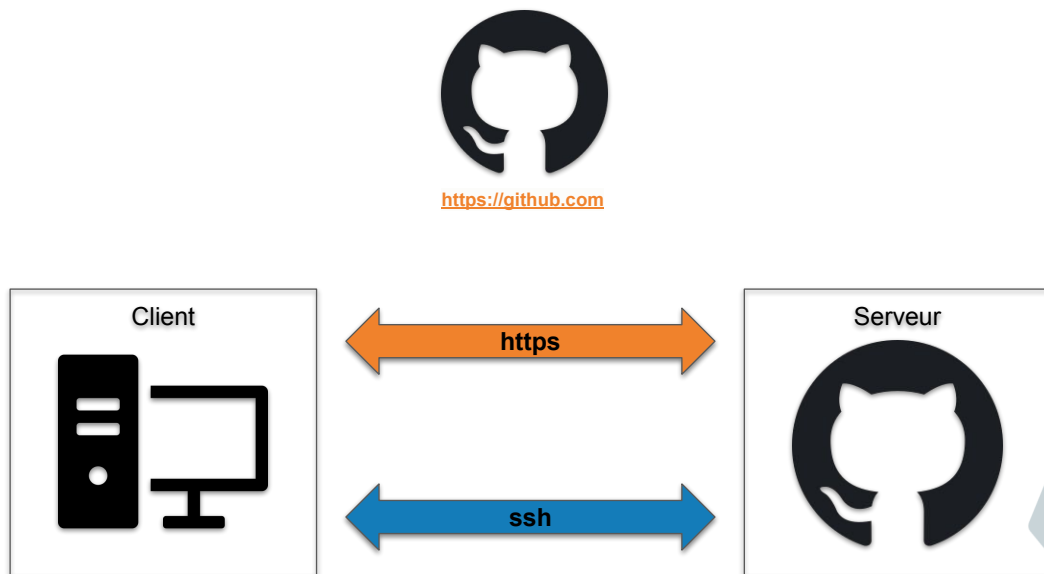
Objectifs

- 1- Faire un fork du dépôt
- 2- Cloner le dépôt qui a été Fork
- 3- Ajouter à la liste des participant votre pseudo ou nom
- 4- Push les modifications sur GitHub
- 5- Faire un pull request

Alde:

- 1- Duplique (fork) le dépôt distant depuis l'interface web. Cela lui permet d'avoir un dépôt distant avec les accès en écriture.
- 2- Cloner sa copie du dépôt distant en local.
- 3- Ajouter un remote vers le dépôt d'origine (celui en lecture seul). Cela permet de se mettre à jour
- 4- Créer une branche de développement en local pour sa contribution
- 5- Faire ses commits et push sur son dépôt distant
- 6- Envoyer un pull request, depuis l'interface web, au propriétaire du dépôt d'origine pour qu'il puisse intégrer les modifications dans son dépôt

Bilan



Nous avons vu dans ce chapitre les fonctions de base de github