GitHub

Octocat.

GitHub est un service web d'hébergement et de gestion de développement de logiciels, destiné aux développeurs.

Ce site est développé en Ruby on Rails (framework) et Erlang (langage de programmation). Il repose sur Git, un système de gestion de code open source créé par Linus Torvalds dans le but d'accélérer le développement d'un logiciel. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki* pour chaque projet.

*(Un wiki est une application web qui permet la création, la modification et l'illustration collaboratives de pages à l'intérieur d'un site web. Il utilise un langage de balisage et son contenu est modifiable au moyen d'un navigateur web).

Fonctionnalités

GitHub est centré vers l'aspect social du développement.

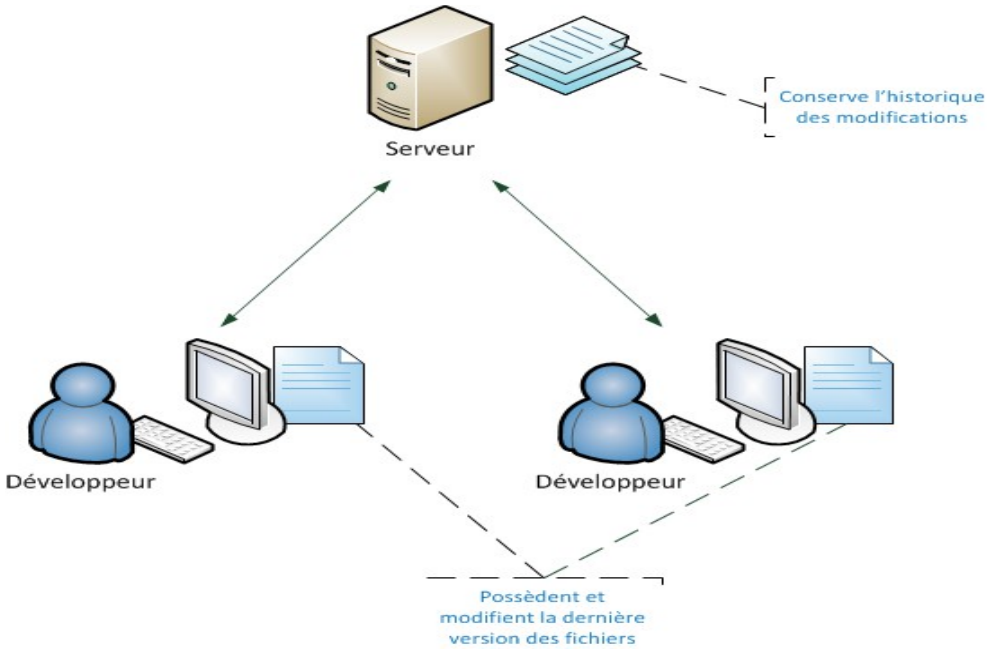
En plus d'offrir l'hébergement de projets avec Git, le site offre de nombreuses fonctionnalités habituellement retrouvées sur les réseaux sociaux comme les flux, la possibilité de suivre des personnes ou des projets ainsi que des graphes de réseaux pour les dépôts (en anglais *repository*). GitHub offre aussi la possibilité de créer un wiki et une page web pour chaque dépôt.

Le site offre aussi un logiciel de suivi de problèmes.

GitHub propose aussi l'intégration d'un grand nombre de services externes, tels que l'intégration continue, la gestion de versions, badges, chat basés sur les projets, etc.

Les documentations des projets sont écrites en langage Markdown (fichiers .md).

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.



Le contrôle de version aide les développeurs à suivre et à gérer les modifications apportées au code d'un projet logiciel. Au fur et à mesure qu'un projet logiciel prend de l'ampleur, le contrôle de version devient essentiel.

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.

Étapes pour déposer un projet

Étape 1 : Initialiser un dépôt (projet) Git

Créer un dossier pour le futur projet

Dans le terminal, depuis le dossier utilisateur (home) :

```
cd Desktop/mon-super-projet
git init
Initialized empty Git repository in C:/Users/conta/Desktop/tests/.git/
```

Vérifier le nouvel état du dépôt

Pour voir comment Git signale sa prise en charge des ajouts, on peut vérifier l'état du dépôt :

```
git status
```

Étape 2 : Ajouter les premières modifications à l'index de Git

Créer un premier fichier

touch index.html && notepad index.html

Vérifier l'état courant du dépôt

Git s'est aperçu qu'il y a un nouveau fichier. Il faut bien comprendre que ce qui intéresse Git ce sont les modifications : autrement dit un nouveau fichier est une modification et c'est pour cela qu'il réagit.

Ajouter le fichier à l'index de Git

Dès que le fichier est ajouté à l'index de Git, ce dernier suivra ses modifications (ajout/édition/suppression de code). Pour Git, ce sont les lignes qui compte : si un seul mot est changé, il considérera que toute la ligne a été modifiée.

```
git add index.html
```

Vérifier à nouveau l'état du dépôt

Cela permettra de constater que Git a effectivement ajouté votre fichier à son index et qu'il est prêt à sauvegarder toutes les modifications que vous effectuerez.

Étape 3 : Faire son premier commit

Un commit est une sauvegarde de modification(s) : lorsque le code a été modifié et que tout marche comme sur des roulettes, il faut faire une nouvelle sauvegarde pour **geler** le projet.

Après modification du code du projet, c'est l'heure de sauvegarder !

La commande Git commit est accompagné d'un tag permettant de coller un bref message au commit.

Étape 4 : Lier son dépôt local à un dépôt distant

Création d'un compte sur un serveur compatible

Avant tout, il faut ouvrir un compte sur un serveur qui comprend Git.

Créer un dépôt sur le serveur

Ensuite, il faut créer un dépôt sur le serveur. Pour cela il suffit de suivre les indications qui sont assez intuitives. En général, on dispose d'un bouton +, on choisit d'ajouter un nouveau **repository** et il suffit seulement de lui donner un titre et de valider sans toucher à rien de plus.

Configuration du dépôt local

De retour sur votre ordinateur, dans votre projet, depuis le terminal...

Utiliser la commande git remote pour voir la liste des URLs vers lesquelles Git pourra envoyer (**pousser**) votre projet.

Évidemment, à ce stade aucun remote n'a été configuré. Pour le faire, il suffit d'utiliser la même commande en ajoutant add, l'url et l'alias que vous choisirez. L'alias sert à éviter d'avoir à taper cette url à chaque fois que vous en aurez besoin.

Étape 5 : Pousser le dépôt local vers le dépôt distant

Pour le moment, il suffit de comprendre que Git peut gérer plusieurs branches. Une branche est simplement un enchaînement des commits que vous aurez effectué.

Envoie du code sur le serveur :
Il suffit d'utiliser la commande git push en lui précisant le serveur sur lequel envoyer le code et la branche à envoyer. Pour le serveur, utilisez l'alias que vous avez choisi avec la commande git remote ad.

En avril 2016, GitHub a annoncé avoir dépassé les 14 millions d'utilisateurs et plus de 35 millions de dépôts de projets le plaçant comme le plus grand hébergeur de code source au monde.