GitHub - найбільший веб-сервіс для хостингу IT-проектів і їх спільної розробки.

Веб-сервіс заснований на системі контролю версій Git і розроблений на Ruby on Rails і Erlang компанією GitHub, Inc (раніше Logical Awesome. Сервіс безкоштовний для проектів з відкритим вихідним кодом і (з 2019 роки) невеликих приватних проектів, надаючи їм всі можливості (включаючи SSL), а для великих корпоративних проектів пропонуються різні платні тарифні плани.

Слоган сервісу - «Social Coding» - на російську можна перекласти як «Пишемо код разом». На футболках ж друкують зовсім іншу фразу: «Fork you!» ( «Відгалузилося!»). З одного боку, вона співзвучна з англомовним лайкою і натякає на неформальну атмосферу. З іншого, ці слова нагадують, що створювати нові Форк з Git можна легко і безболісно - традиційно, до створення гілок розробники проектів з відкритим вихідним кодом відносяться негативно - а також співзвучна назві однієї з можливостей GitHub - черги ФОРКОМ.

GitHub також є одним з найбільших онлайн-сховищ (англ) спільної роботи по всьому світу.

# **git**

Git - це система контролю версій, розроблена Лінусом Торвальдсом.

# **Система контролю версій**

Коли розробники створюють новий проект, вони завжди продовжують вносити оновлення в код. Навіть після запуску проектів їм все одно потрібно оновлювати версії, виправляти помилки, додавати нові функції і т. Д.

Система контролю версій допомагає відслідковувати зміни, внесені в базу коду. Більш того, він записує, хто вніс зміни і може відновити стертий або змінений код.

Перезаписаних кодів не існує, оскільки Git зберігає кілька копій в сховище.

# **Hub?**

Концентратор в GitHub - це те, що перетворює командний рядок, таку як Git, в найбільшу соціальну мережу для розробників.

Крім участі в певному проекті, GitHub дозволяє користувачам спілкуватися з однодумцями. Ви можете слідкувати за людьми і дивитися, що вони роблять або з ким вони спілкуються.

# **Репозиторій**

Репозиторій або сховище - це каталог, в якому зберігаються файли вашого проекту. Він може бути розташований в сховище GitHub або в локальному сховищі на вашому комп'ютері. Ви можете зберігати файли кодів, зображення, аудіо або все, що пов'язано з проектом, в сховище.

# **Гілка**

Гілка це копія вашого сховища. Ви можете використовувати гілку, коли хочете зробити розробку ізольовано.

Робота з гілкою не вплине на центральне сховище або інші гілки. Якщо ви зробили цю роботу, ви можете об'єднати свою гілку з іншими гілками і центральним репозиторієм, використовуючи запит на витяг.

# **Команди Git**

-git config

Одна з найбільш часто використовуваних git команд. Вона може бути використана для вказівки налаштувань, таких як електронна пошта, ім'я користувача, формат і т.д.

Наприклад, дана команда використовується для установки адреси електронної пошти:

git config --global user.email [адрес@gmail.com](mailto:адрес@gmail.com)

git init

Ця команда використовується для створення GIT сховища. Приклад використання:

git init

git add

Команда git add може бути використана для додавання файлів в індекс. Наприклад, наступна команда додасть файл під назвою temp.txt присутній в локальному каталозі в індекс:

git add temp.txt

git clone

Команда git clone використовується для клонування сховища. Якщо сховище знаходиться на віддаленому сервері, використовується команда такого роду:

git clone имя.пользователя@хост: / шлях / до / сховища

І навпаки, для клонування локального сховища використовуйте:

git clone / шлях / до / сховища

git commit

Команда git commit використовується для коммітов змін в файлах проекту. Зверніть увагу, що коммітов не відразу потрапляють на віддалений репозиторій. застосування:

git commit -m "Повідомлення йде разом з коммітов"

git status

Команда git status відображає список змінених файлів, разом з файлами, які ще не були додані в індекс або очікують коммітов. застосування:

git status

git push

git push ще одна з часто використовуваних git команд. Дозволяє помістити зміни до головного гілку віддаленого сховища пов'язаного з робочим каталогом. наприклад:

git push origin master

git checkout

Команда git checkout може бути використана для створення гілок або перемикання між ними. Наприклад, наступний код створить нову гілку і переключиться на неї:

command git checkout -b <ім'я-гілки>

Щоб просто переключитися між гілками використовуйте:

git checkout <ім'я-гілки>

git remote

Команда дозволяє користувачеві підключитися до віддаленого сховища. Дана команда відобразить список віддалених репозиторіїв, налаштованих в даний момент:

git remote –v

Ця команда дозволить користувачеві підключити локальний репозиторій до віддаленого сервера:

git remote add origin <адрес.удаленного.сервера

git branch

Команда git branch може бути використана для відображення, створення або видалення гілок. Для відображення всіх існуючих гілок в репозиторії введіть:

git branch

Для видалення гілки:

git branch -d <ім'я-гілки>

git pull

Команда pull використовується для об'єднання змін, присутніх в віддаленому репозиторії, в локальний робочий каталог. застосування:

git pull

git merge

Команда git merge використовується для об'єднання гілки в активну гілка. застосування:

git merge <ім'я-гілки>

git diff

Команда git diff використовується для виявлення відмінностей між гілками. Для виявлення відмінностей з базовими файлами, використовуйте

git diff --base <назва\_пакунка>

Наступна команда використовується для перегляду відмінностей між гілками, які повинні бути об'єднані, до їх об'єднання:

git diff <гілка-джерело> <гілка-мета>

Для простого відображення існуючих відмінностей, використовуйте:

git diff

git tag

Використовується для маркування певних коммітов за допомогою простих міток. Прикладом може бути ця команда:

git tag 1.1.0 <вставте-commitID-тут>

git log

Запуск команди git log відобразить список всіх коммітов в гілці разом з відповідними відомостями. Приклад результату:

commit 15f4b6c44b3c8344caasdac9e4be13246e21sadw

Author: Alex Hunter <alexh@gmail.com>

Date: Mon Oct 1 12:56:29 2016 -0600

git reset

Команда git reset використовується для скидання індексу і робочого каталогу до останнього стану коммітов. застосування:

git reset --hard HEAD

git rm

git rm використовується для видалення файлів з індексу і робочого каталогу. застосування:

git rm імяфайла.txt

git stash

Можливо одна з найбільш маловідомих команд git. Вона допомагає у збереженні змін на тимчасовій основі, ці зміни не потраплять в комерц відразу. застосування:

git stash

git show

Для перегляду інформації про будь-якому git об'єкті використовуйте команду git show. Для прикладу:

git show

git fetch

git fetch дозволяє користувачеві доставити всі об'єкти з віддаленого сховища, які не присутні в локальному робочому каталозі. Приклад застосування:

git fetch origin

git ls-tree

Команда git ls-tree використовується для перегляду дерева об'єкта разом з назвою, режимом кожного предмета і значенням SHA-1. Наприклад:

git ls-tree HEAD

git cat-file

Використовуйте команду git cat-file, щоб переглянути тип об'єкту за допомогою SHA-1 значення. наприклад:

git cat-file -p d670460b4b4aece5915caf5c68d12f560a9fe3e4

git grep

git grep дозволяє користувачеві проводити пошук фраз і слів у вмісті дерев. Наприклад, для пошуку www.hostinger.ru у всіх файлах використовуйте цю команду:

git grep [www.hostinger.ru](http://www.hostinger.ru)

gitk

gitk - це графічний інтерфейс локального сховища. Викликати його можна виконавши дану команду:

gitk

git instaweb

За допомогою команди git instaweb можна запустити веб-сервер, пов'язаний з локальним репозиторієм. Браузер також автоматично перенаправлятиметься на нього. наприклад:

git instaweb -httpd = webrick

git