**Git і GitHub**

Git - це одна з систем контролю версій. По суті це означає, що вона зберігає всю історію змін проекту. Історія Вашого проекту і історія змін цього ж проекту у Ваших колег - у всього буде копія. Це повна протилежність SVN, де вся історія змін зберігатися в одному місці.

GitHub, часто плутають з Git. Насправді - це хостинг репозиторіїв. Можливо Вам поки незрозуміло що таке репозиторій, але не поспішайте закривати статтю, до кінця все проясниться. Якщо коротко, GitHub - це те місце, куди Ви будете завантажувати історію змін проекту використовуючи Git.

**Шпаргалка**

Використовуючи Git Bash або Terminal перейдіть в кореневу директорію проекту. Якщо Ви використовуєте Git Bash, то за допомогою правого кліка можна вибрати "Git Bash Here" і він запуститься в робочій директорії.

**git init**

Ця команда створить .git репозиторій у Вашому проекті. Репозиторій або "repo" це колекція всіх змін, які були здійснені протягом усього часу після ініціалізації сховища. Це перше що потрібно зробити для нового проекту.

**git config --global user.name "Ваше ім'я"**

**git config --global user.email "ВашаПочта@mail.com"**

Ці команди визначать інформацію, яка буде використовуватися при кожному commit (фіксування змін). Їх варто виконати всього один раз при первинній установці Git.

**git add імяФайла.розширення**

Замініть "імяФайла.розширення" на будь-який файл, зміни якого Ви намагаєтеся зафіксувати, наприклад "index.html". Ця команда додасть файл в "staging area" (ділянка підготовки). Сприймайте staging area, як секцію в якій файли проходять підготовку до переміщення в Ваш репозиторій.

**git add .**

Якщо Ви хочете додати все з директорії проекту в staging area, то ця команда зробить все сама.

**git add \*.html**

Якщо Ви хочете додати всі файли з розширенням .html в staging area то ця команда відмінно підійде. Розширення можна змінювати в залежності від уподобань.

**git status**

Покаже що вже було додано в staging area і які файли були змінені і чекають переміщення в staging area.

**git reset імяФайла.розширення**

Прибирає обраний файл з staging area.

**git rm --cached імяФайла.розширення**

Прибирає файл з staging area і визначає його як ігнорований.

**git commit -m "Опис коммітов"**

Бере файли з staging area і "фіксує" їх в Ваш локальний репозиторій. У лапки слід вставити короткий опис змін для конкретного коммітов. Постарайтеся описати Комміт короткими деталями, наприклад: "усунув проблему при зміні імені користувача" замість подібних повідомлень "якісь зміни"

**touch .gitignore**

Ця команда створить файл з назвою .gitignore. Ви можете відкрити цей файл у текстовому редакторі і прописати назви файлів або директорій, зміни в яких ви не хотіли б відслідковувати (вони будуть ігноруватися Git). Зміни в ігнорованих файлах не відобразяться виконанні git status.

**git branch назваГілки**

Створює сутність, звану branch (гілка). Гілка - це точна копія Ваших файлів.

**git checkout "назваГілки"**

Дозволить Вам перемкнути контроль над створеною Вами гілкою і працювати в її межах. Тут Ви можете здійснювати будь-які зміни коду. Коли Ви будете готові можна зробити commit змін і відправити зміни в GitHub (про це нижче) або ж можна видалити гілку, якщо щось пішло не так або Вам більше не потрібні зміни зроблені в цій гілці.

**git merge назваГілки**

Перебуваючи в Master (головної) гілки, Ви можете використовувати цю команду, щоб взяти коммітов з будь-якої з гілок і з'єднати їх разом.

**git remote add origin https://github.com/імяКористувача/проект.git**

Ця команда визначить "місце розташування" Вашого віддаленого сховища. Все що було до цього відбувалося виключно в локальному репозиторії на Вашому комп'ютері. Вам потрібно буде перейти в GitHub аккаунт і створити новий віддалений репозиторій, куди Ви зможете відправляти зміни з локального сховища. Після створення віддаленого сховища вам буде надано посилання, і це посилання буде місцем, яке ви хочете використовувати в наведеній вище команді.

**git remote**

Виведе список з усіх віддалених репозиторіїв, які були додані до Вашого проекту.

**git push -u origin master**

Ця команда відправить локальні зміни в віддалений репозиторій. Таким чином цю команду варто прописувати тільки перший раз.

**git push**

Це те, що ви будете використовувати для надсилання коду до GitHub після початкового натискання.

**git clone https://github.com/імяКористувача /проект.git**

Якщо у Вас відсутній проект на особистому або робочому комп'ютері, то ця команда допоможе клонувати / завантажити весь проект в поточну директорію.

**git pull**

Якщо Ви працюєте над одним і тим же проектом з декількома людьми, то ця команда дозволить завантажити останню версію з віддаленого сховища та оновити вашу локальну версію.

**git diff**

Показує всі зміни які відбулися у ваших файлах

**git log**

Показує список комітів

**git show idCommit**

Показує що ми змінили при даному коміті