

Работа в git

pwd — покажи в какой я папке;

ls — покажи файлы в папке, где я сейчас;

cd first-project — перейди в папку first-project;

cd first-project/html — перейди в папку html, находящуюся в папке first-project;

cd .. — перейди на уровень выше в родительскую папку;

cd ~ — перейди в домашнюю директорию (у нас это /Users/stas\_basov);

mkdir second-project — в текущей папке создай папку с именем second-project;

rm about.html — удали файл about.html;

rmdir images — удали папку images;

rm -r second-project — удали папку second-project и всё, что она содержит;

touch index.html — создай файл index.html в текущей папке;

touch index.html style.css script.js — если нужно создать несколько файлов, их имена можно вводить через пробел.

Настройка git

git config - это конфиг

git config --global -это опция вода данных

git config --global user.name"Ваше имя" - мои даные туту мо имя

git config --global user.email"емэил" - мои данные мой емаил

git config --list - чек лист всех конфиго что у нас есть и что вних.

Подключение git к проекту

git init - пишем в папки проекта и тогда git начнёт отслеживать эту папку.

git status - выведит какеи файлы и папки отслеживает наш git в этом проекте

git add и файл - зделает файл готовым к сохранению.

git add --all или git add -A - выберит все файлы и подготовит их к сохранению

git commit -m "название комита" - чтобы сделать комимт не забудьте подготовить файлы

git log - покажет историю комитпше ов

Подключение git к gitHab

1.SSH КЛЮЧЬ

Правила обычно ключь SSH лежит домашней дириктории к не перейти cd ~

ls -la .ssh/ - чтобы найти ключь но его может и не быть

ssh-keygen -t ed25519 -C "potili@bk.ru" генирируем ключь их два локальный и пуьличный и пишем пароль

ssh-agent и eval $(ssh-agent -s) - на ищет SSH-ключ на вашем компьютере

ssh-add ~/.ssh/id\_ed25519 - привязываем

clip < ~/.ssh/id\_ed25519.pub - копируем

2.Связываем локальный и удалённый репозитории

создаём репозиторий

копируем адрес SSH в репозитории

git remote add origin git@github.com:YandexPracticum/first-project.git - подключаем удалёный и локальный

репозиторий для этого нужно зайти в папку с проектом и прописать там это но только заменить ключь SHH на свой

git branch -M main - нужно потомучто по умолчанию ветка называется master а gitHab работает

с название main и всё каманды зачиклены на нём так что надо

git push -u origin main - связывает проексто светкой

3.Синхронизация локального и удалённого репозитория с gitHiub

git status - покажет какие файлы не готовы к комиту если в них есть изменения горят красным

git add -A - выбор всех файлов для комита горят зелёным

git commit -m "название комита" - создаём комит

git push - отправить изменения, или просто «запушить».

git pull - проверка нового обновления в проекте если я не один работаю над проектом

git clone git@github.com:maik791277/how-to-learn.git - клонироване новго репизитория

Например, находясь в папке dev начните вводить cd first и нажмите Tab.

Если папка first-project есть внутри dev, командная строка

автоматически подставит её имя. Останется только нажать Enter.

Шпоргалка

Команды для синхронизации локального репозитория с удалённым.

git remote add origin https://github.com/YandexPracticum/first-project.git — находясь в папке с локальным репозиторием, привязываем его к удалённому (URL у вас будет свой);

git push -u origin main — заливаем все файлы из локального репозитория в удалённый, который уже привязали.

Команды для того чтобы сделать сохранение — коммит.

git add название\_файла — готовим выбранный файл к коммиту;

git add -A — чтобы ничего не потерять, можно подготовить к коммиту сразу все файлы, в которых были изменения;

git commit -m "комментарий к коммиту" — делаем коммит. К сохранённым файлам оставляем комментарий для того, чтобы проще было понять, какие сделаны изменения.

Команды для публикации изменений:

git pull — забрать изменения, сделанные другими разработчиками;

git push — опубликовать изменения в удалённый репозиторий.

горячие клавиши

переходит в начало строки Ctrl + A;

возвращается в конец строки Ctrl + E;

Команды в терминале необязательно вбивать и выполнять по очереди. Команды можно указывать не по одной, а сразу списком. Для этого их разделяют двумя амперсандами &&:

Конфликты в Git

git commit -m "Change z-index, fix navigation links in header" && git checkout main  Теперь вы их коммитите и переходите в ветку main:

git merge bugfix/header В ветке main вы пытаетесь смержить её с bugfix/header, но вместо коммита слияния видите:

Команда git status покажет untracked-файлы в этой рабочей области, флаг -sb — краткую версию информации:

Команда git diff покажет изменения во всех файлах:

Затем добавляем изменённые файлы в staged и делаем коммит:

git add -A && git commit -m "Resolved conflict in main"