GIT

**Основы**

**init –** инициализация уже существующего проекта

**add –** указиваем индексированые файлы (которые нужно добавить под версионный контроль)

**\* -** рекурсивно добавить всё

**- A —** добавить все файлы в деректории

**clone –** клонирование репозитория

*Автоматически настраивает лок. ветку «master» на отслеживание удалённой ветки «master» на сервере*

**status** – определение состояния файлов

**-s ( --short ) –** краткая сводка

**diff —** показывает что конкретно поменялось

**– no arg -** что изменили но не проиндексировали

**-- cached -** что проиндексировали и собираетесь фиксировать

**branch [b-name] –** создать новую указывающую на текущий HEAD

*метка HEAD указывает на ту ветку где мы работаем*

- **no args –** посмотреть список всех веток

**-d –** удалить ветку *( не позволит удалить ветку с наработками )*

-**D –** удалить ветку с наработками

**-v –** последний коммит из каждой ветки

**--merged –** ветки которые уже слили с текущей выбраной

**--no-merged –** ветки которые не слиты с текущей

**cherry-pick –** прим. изменения к дереву проекта, которые были внес. комит.

*( перенести коммит из одной ветки в другую )*

***Перед этим нужно созать ветку и перейти в неё***

**-n –** не создавать коммит при переносе изменений

**-continue –** продолжить после разрашения конфликтов

**[hash-commit-1] -** хэш коммита который был лишним

**[hash-com-beg]..[hach-com-end] –** перенести несколько подряд

**[ветка] –** когда необходимо перенести последний ком. ветки

**checkout [b-name] [удал. ветка.] –** переключиться на ветку *(доп. Создаст ветку на основе удал. ветки)*

**-b –** создать и сразу переключиться на ветку

**--track [уд. срв./ветка]–** альтернатв. **checkout –b [ветка] [уд.срв./ветк]**

**merge [b-name] –** слить изменения в определённую ветку

**n:** $ git checkout master

$ git merge hotfix

**rebase [ветка] –** взять все измения на одной ветке и потворить их на другой

**commit —** фиксация своих изменений

-- **no arg** **-** откроет редактор для ввода коментария

**-v —** добавляет к коментарию вывод команды **«diff»**

**-m** — свой коментарий к комиту

**-a —** говорит гиту автоматически всё проиндексировать

**rm —** удалить файл из гита ( удалить из нашего индекса и нашего каталога )

*( если просто удалить будет: «Изменён но не обновлён»)*

**-f —** принудительное удаление для проиндексированого файла

--**cached —** удалит файл из индекса(репозитория) оставив на диске пользователя

**mv —** переименование / перемещение файла

**mergetool –** графичиское разрешение конфликтов

**Просмотр истории коммитов**

**log —** список коммитов данного репозитория в хрон. Порядке

**-p -**  показать разницу ( **«diff»** ) по строкам

**--word-diff —** разница по словвам

**--stat —** краткая статистика по коммитам

**--prety —** позволяет изменять формат вывода лога

**=oneline —** каждый коммит в одну строку

**=format: -** полностью собственный формат лога

**--graph —** маленький ASCII-граф показывает историю веток

**--author —** фильтрация по автору

**--grep —** по ключевым словам в сообщении

**--all-match —** удовлетворяют и первому и второму критерию

**-<n>** — колличество последних коммитов

**-since=2.weeks —** коммиты за последние две недели

**-- [ folder / file ] -** коммиты для этого файла / папки

**Отмена изменений**

**commit --amend —** изменение последнего коммита

**n:** $ git commit -m 'initial commit'

$ git add forgotten\_file

$ git commit --amend

**reset HEAD <file> -** отменить индексацию файла ( не попадёт в коммит )

**--hard [hash-commit-0] –** сместить указатель на *hash-commit*

**checkout -- file —** вернуть к начальному состоянию

**Робота с удалёнными репозитроиями**

*удалённый сервер = удалённый репозиторий = сокращение*

**remote —** перечисляет список имён удаллёных репозиториев

**-v —** какому URL соответствует сокращенное имя

**add [сокращени] [url]—** добавить удал. репоз. под своим именем

**show [сокращение] —** информация об удал. репозитории

**rename [oldName] [newName] —** изменить сокращеное имя

**rm —** удалить ссылку на удал. репозиторий

**fetch [имя удал. репоз.] -** связываеться с удаллёеным репозиторием и забир. все данные проекта которых у вас нет.

**n:** git fetch origin ↓

*Команда забирает данные в ваш локальный репозиторий но не сливает их с вашими наработками. Необходимо вручную*

**pull —** автоматически извлекает и сливает данные с удал. репозитория

**push [удал. репоз.] [ветка] -** отправить изменения на репозиторий

**--tags —** отправляет все метки на удаллёный репозиторий

**[уд. сервер] :[ветка] –** удалить ветку на удал. сервере

**Робота с метками**

**tag —** виводит метки

**-l —** поиск по шаблону

git tag v1.4-lw — **легковесная метка**

**-a —** создание аннотированной метки

**-m —** коментарий

-**control sum —** для отметки коммита

**.gitignore**

1. Можно начать шаблон символом слэша (/) чтобы избежать рекурсии.
2. Можно заканчивать шаблон символом слэша (/) для указания каталога.
3. Можно использовать стандартные glob шаблоны.
4. **! =>** показывает что нужно отслеживать