**Основы git**

1 Команды Linux(Bash)

1. Чтобы просмотреть все папки в директории пишем команду (ls) или команда (dir)
2. Чтобы перейти в другую директорию пишем (cd + <путь>), путь можно полностью не дописывать, а нажать (Tab)
3. Чтобы выйти на папку уровнем выше (cd..)

3.1) Чтобы выйти на два уровня выше (cd../..)

1. Создать директорию команда (md->Windows)

(mkdir->Linux)

1. Чтобы создать файл (touch test.txt)
2. Скопировать один файл в другой (cp test.txt test\_copy.txt)
3. Чтобы переименовать файл (mv test.txt test\_new.txt)
4. Вывести что либо в файл (echo «Hello" > text.txt)
5. Открыть файл в терминале (cat test.txt)
6. Удалить файл (rm test.txt)
7. Удалить директорию (rm –R testDir)

2 Начальная настройка репозитория

1. Заходим в папку с файлами и используем команду для инициализации репозитория (git init)
2. С помощь просмотра расширенного поиска скрытых папок (ls -a) можем увидеть папку (.git)
3. Команда для проверки статуса репозитория (git status)
4. Устанавливаем связь с репозиторием на github, где связь назовём (origin) (git remote add origin <http://path>)
5. Проверяем связь с репозиторием (git remote –v)

3 Commit

1. git status (проверить статус репозитория)
2. git add <файл> (Добавить файл в commit)
3. git rm --cached <файл> (Убрать отслеживание файла, который уже в репозитории есть)
4. git rm <имя файла> удалить файл из репозитория локального и из github
5. git commit (Чтобы внести изменения)(Нажимаем один раз (i), чтобы сделать коммент, потом (ESC), потом (shift) + два раза (z))
6. git commit –m ‘Start’ (Сделать commit не заходя в текстовый редактор)
7. git commit –am ‘modify’ (Создать commit внеся файлы, которые были изменены, если они до этого существовали)
8. git commit –a (Создать commit, но для ввода комментария перебросит в текстовый редактор)
9. git restore <имя файла> убрать изменения файла из директории
10. git restore --staged <файл> (Убрать отслеживание файла из commit)

4 geetkeep, gitgnore

1. touch <имя директории>/.gitkeep (Создаём в директории файл .gitkeep , чтобы папка была видна для git)
2. ls –a <Имя директории> (Чтобы посмотреть, что находится в директории не заходя в неё)
3. .gitignore (Всё что находится в данном файле будет игнорироваться для добавления в commit)

5. Создание веток

1. git branch (Просмотреть доступные ветки)
2. git branch <имя ветки> Создание новой ветки
3. git checkout <имя ветки> Перейти на новую ветку
4. git log Просмотреть все коммиты
5. git log - - oneline Вывести историю коммитов одной строкой на каждый коммит
6. git branch <имя новой ветки> <имя ветки от которой ответвлимся> (Если хотим ответвляться не от той ветки на которой находимся, а от любой другой)

6.Удалённые репозитории

1. ssh-keygen (Создаём ключ для репозитория, вывести его содержимое можно командой cat, а потом загрузить на github)
2. clear (Очистить консоль)
3. git clone <путь до репозитория> <Папка куда надо клонировать> Клонируем репозиторий
4. git push –u origin <Имя ветки> (Отправляем изменения в репозитории, и создаём ветку, если её нет)
5. git pull (Выкачиваем все изменения для ветки на которой находимся)
6. git fetch (Загрузить все изменения с репозитория с новыми ветками и всеми изменениями)
7. Слияние веток
8. После использования git fetch нельзя увидеть ветки, которые не были созданы локально, их надо явно вызвать git checkout <имя ветки>
9. git merge <Имя ветки> объеденить ветку с веткой на которой сейчас находимся
10. git branch –D <имя ветки> удаляем ненужную ветку только из локального хранилища
11. git push - - delete origin <имя ветки> удаляем ветку из репозитория на github
12. git add . загрузить все файлы в commit
13. git rebase <имя ветки откуда возьмём файлы> начало ветки на которой мы сейчас находимся становится последним коммитом(и далее) ветки которую мы взяли за основу
14. git push --force принудительно загружаем новую ветку в репозиторий
15. Если создать локальный репозиторий раньше репозитория на github, то чтобы инициализировать ветку master используем (git pull <путь до репозитория>)
16. Git checkout –orphan <имя ветки> создаём пустую ветку, затем удаляем из неё все файлы git rm –rf . (не забываем про точку на конце второй команды)
17. Управление версиями

1. git tag посмотреть версии проектов
2. git tag <имя версии> дать коммиту на котором сейчас находимся номер версии
3. git push - -tags (загрузить коммит с тегами в git)
4. git tag –d <номер версии> удаляем тэг
5. git push - - delet origin <имя тэга> удаляем тэг из репозитория на git
6. Создание pull-request
7. Заходим на сайт github делаем pull-request для веток, одна из которых наследовалась от другой