**Часть 1.** Установка, запуск, настройка.

1) Установить git -> <https://git-scm.com/downloads>.

Устанавливать можно с рекомендуемыми параметрами, кроме текстового редактора. Там по умолчанию стоит vim, лучше поменять на обычный блокнот или Notepad++ (установить его заранее, <https://notepad-plus-plus.org/downloads/>).

2) Выбрать папку(пустую обычно) для проекта.

3) Правой кнопкой по пустоте в этой папке, выбрать "Git Bash Here".

Далее в консоли:

4) git init (создает систему контроля версий в данной папке).

5) Добавить сетевой репозиторий "Pascal\_lectures"

git remote add origin https://github.com/AleksandrVakulenko/Pascal\_lectures.git

*(это одна строка)*

Ctrl+C, Ctrl\_V НЕ РАБОТАЮТ В КОНСОЛИ, РАБОТАЮТ Ctrl+Insert и Shift+Insert.

* Настройки имени пользователя:

git config --global user.name "FIRST\_NAME LAST\_NAME"

git config --global user.email "MY\_NAME@example.com"

Папку с созданным git можно переименовать или переместить, ему нет дела до этого.

Для удаления git просто удалите скрытую папку ".git"

**Часть 2.** Работа в консоли для получения файлов из удаленного репозитория.

1) Получить актуальное состояние ветки master из репозитория origin в ту локальную ветку, которая в данный момент активна.

git pull origin master

**Часть 3.** Файл gitignore.

Для того, чтобы git не интересовался лишними файлами - в текстовом файле .gitignore (точка перед именем и без расшерения txt на конце!) перечислены маски имен файлов, которые нужно игнорировать.

Для проектов Delphi pascal это:

\*.exe

\*.bdsproj

\*.local

\*.cfg

\*.~1~

\*.~2~

**Часть 4.** Добавление файлов

В результате работы с проектом некоторые файлы изменились/удалены/созданы. Чтобы зафиксировать этот результат выполняется следующая последовательность:

1) git add .

Команда add добавляет файлы в проект контроля версий, точкой(.) обозначено "все, что найдешь".

При необходимости можно добавлять не все файлы, а только необходимые, но обычно нужды в этом нет.

2) git commit -m 'some text about this commit'

git commit -m "some text about this commit"

Именно команда "commit" создает "точку сохранения" проекта.

Параметр "-m" это сокращение от message и означает, что далее ожидается строка текста сообщения.

Символы " и ' равнозначны, но должны быть одинаковые слева и справа.

После выполнения этой команды точка сохранения готова, но находится только в локальном репозитории.

**Часть 5.** Передача состояния в удаленный репозиторий.

В случае коллективной работы с удаленным репозиторием необходимо отправить туда список всех коммитов, которые были созданы за текущий период активности разработчика.

Отправлять разом можно сколько угодно созданных локально коммитов.

Логично это делать каждый раз после конца рабочей активности имеющей логическое завершение хотя бы по какому-то вопросу.

Выполняется командой:

git push origin master

где origin - удаленный репозиторий, куда отправляются изменения.

master - в какую ветку удаленного репозитория отправить изменения.

Изменения отправляются из текущей выбранной ветки

**Часть 6.** Сводка последовательностей команд для различных ситуаций.

* *Настройка имени и e-mail (один раз после установки git).*

git config --global user.name "FIRST\_NAME LAST\_NAME"

git config --global user.email "MY\_NAME@example.com"

* *Создание нового репозитория.*

git init

git remote add origin https://...

* Создание коммита.

git add .

git commit -m ‘some message’

git push origin master

* Отправка коммитов в сетевой репозиторий.

git push origin master

* Получения коммитов из сетевого репозитория.

git pull origin master

Как сделать красивые логи.

Изображение выглядит как текст

Автоматически созданное описание

В папке C:\Users\{your user name}\

* Найти файл .gitconfig
* Добавить в него содержимое for\_git\_config.txt
* Сохранить

Вызывать команды:

git lg1

git lg2

git lg3

Какая больше нравится.