Язык разметки Markdown (.md)

# Самый главный заголовок

## Заголовок второго уровня и так до шестого уровня

Чтобы текст написать нужно просто писать текст в пустую строку

Чтобы вставить Ссылку, нужно написать [текст ссылки](Ссылка)

Чтобы вставить картинку, нужно написать ![описание картинки](Ссылка)

Списки:

* Нумерованный через точку (1. Пункт списка/ 2. Пункт списка/)
* Нумерованный через тире (– пункт списка)

**Система контроля версий Git** – самый популярный в мире инструмент для командной работы над IT проектами

**Git решает следующие задачи:**

* Хранение актуальных версий файлов
* Отслеживание изменений файлов
* Командное обсуждение изменений
* Параллельное развитие проекта
* Возможность копирования проекта
* Публичная и приватная демонстрация файлов

**Перед стартом работы с Git необходимо:**

1. Настроить видимость скрыты папок и файлов
2. настроить видимость расширений файлов
3. Настроить работу редактора кода
4. Настроить плагин для универсального языка разметки Markdown

**Командная строка (терминал)** – это интерфейс для общения с компьютером напрямую, без дополнительных программ

**Создание локального репозитория. Последовательность действий:**

* создайте папку проекта на компьютере
* откройте для папки терминал
* выполните команду **git init**
* проверьте, что появилась папка **.git**

**Чтобы сделать снимок проекта (сформировать коммит):**

* добавьте файл или файлы в отслеживаемые командой **git add имя файла**
* создайте коммит и подпишите его командой: git commit -m "подпись проекта"

**Чтобы проверить состояние файла:**

* на любом из этапов вы можете выполнить команду **git status**

**Путешествие по истории проекта:**

* смотреть историю можно командой **git log с разными флагами --oneline или --all**
* Исправить что-то в последнем коммите, который хранится только на компьютере, можно при помощи **флага --amend**
* Чтобы не добавлять в коммит все файлы по именам можно, написать **git add и флаг -А**
* Переключить между коммитами проекта можно командой **git checkout хеш**. Вернуться к последнему коммиту можно командой **git checkout -main**

**Github**

SSH ссылка [git@github.com:Wrdalf/test.git](mailto:git@github.com:Wrdalf/test.git)

а **GitHub** – самый популярный в мире сервис для публикации и хранения в интернете проектов, а также для командной разработки.

а **SSH-ключ** – это уникальный набор символов, который позволяет связать локальный и удаленный репозитории, чтобы:

* из командной строки отправлять данные прямо в удаленный репозиторий
* проекты попадали в нужный профиль, а не чужой

**форк (fork)** позволяет создать копию опубликованного на gitHub репозитория и работать с ним самостоятельно. При создании форка репозиторий копируется в вашу учетную записать GitHub. Это позволяет вносить изменения в свою копия проекта без ущерба для основного репозитория

**Ветка в Git** – опция, которая позволяет создавать «параллельную реальность» в проектах, когда:

* над проектом работают несколько человек
* каждый делает свою задачу
* чтобы не мешать друг другу, каждый работает в своей ветке
* Когда задача готова, то все сливается в основную ветку

**ключевые действия и команды:**

* Чтобы с**оздать копию чужого проекта на GitHub**, нужно нажать кнопку **Fork** на странице проекта
* Чтобы **клонировать удаленный репозиторий на компьютер**, нужно скопировать ссылку по кнопке Code и затем выполнить в терминале команду **git clone ссылка-на-репозиторий**. после клонирования не забыть перейти в новую папку командой **cd имя-папки**
* Чтобы создать новую ветку и переключиться на нее, нужна команда giе checkout -b имя-новой-ветки
* Чтобы **переключиться между ветками**, выполняем команду **git checkout имя-ветки**
* для слияния находимся на ветке, куда сливаем, и выполняем g**it merge имя-сливаемой-ветки**
* При **разрешении конфликта** обязательно удаляем служебные символы, оставляем нужный вариант и коммитим изменения
* чтобы сгенерировать SSH-ключ, нужно в терминале ввести команду **ssh keygen**, скопировать окончание пути ключа. Далее в терминале введите команду **cat ~ окончание пути** до файла, который вы только что скопировали. Скопируйте длинный набор символов. В GitHub нажмите кнопку **New SSH Key** и вставьте данный набор символов
* Чтобы создать новый репозиторий, нажмите кнопку New Repository
* Чтобы связать удаленный и локальный репозиторий, выполните команду **git remote add origin ссылка на папку удаленного репозитория**
* Чтобы отправить что-нибудь в удаленный репозиторий, нужно воспользоваться командой git push -u origin main

**Командная работа в Git и GitHub. Часть 1**

**.gitignore**

* Файл-настройка. Создается в папке проекта. В нем указывается, какие файлы нужно игнорировать при публикации на GitHub
* Имя файла должно начинаться с точки, а в конце не должно быть никакого расширения
* Правила игнорирования сработают только для файлов, не добавленных в индекс, неотслеживаемых, до команды git add

**При командной разработке важно уметь:**

* отправлять изменения в проект в удаленный репозиторий
* работать с коммитами
* настраивать видимость файлов и папок

о .gitignore – это файл-настройка, который позволяет игнорировать нужные папки и файлы по время публикации в удаленный репозиторий на github

**Ключевые действия и команды при командной разработке**

* Для обновления командного проекта на GitHub нужно использовать команду **git pull**
* Чтобы переключаться к предыдущему состоянию проекта, используйте команду **git checkout хеш коммита**
* Посмотреть всю историю проекта можно через команду **git log --oneline --all**
* Чтобы отменить коммит в удаленном репозитории, нужно использовать команду **git revert хеш коммита для отмены**
* Настроить . gitignore можно для папки и файла с помощью команды **git rm --cached имя-файла**