

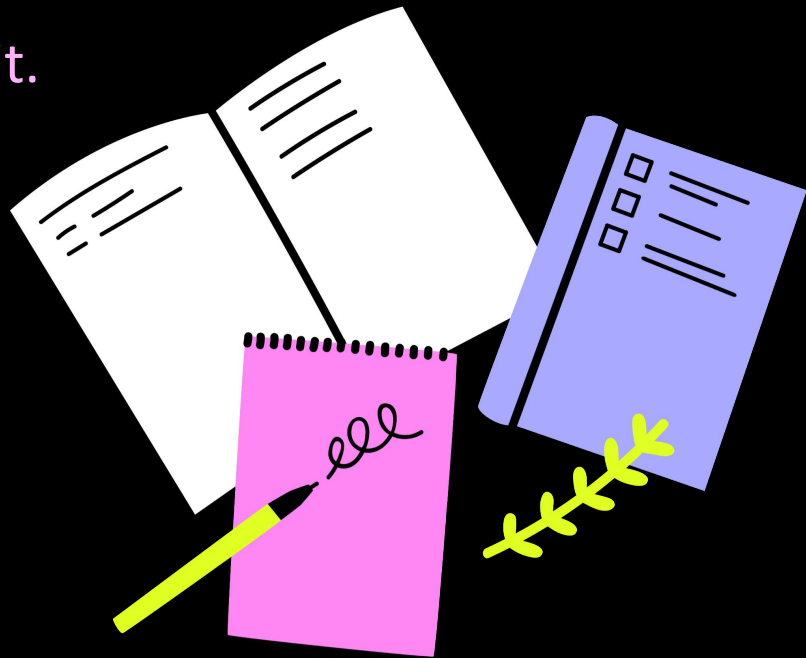


Введение в контроль версий

Семинар 1

Знакомство с контролем версий Git.

Настройка, основные команды.



Давайте знакомиться!



Свищев Алексей

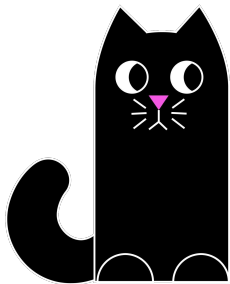
Преподаватель C#

- ⚡ Разработка плагинов QGIS для определения лесных массивов и водоемов;
- ⚡ Разработка парсеров и баз данных;
- ⚡ Работал в онлайн-школе Kodland;
- ⚡ Доношу до учеников любую информацию

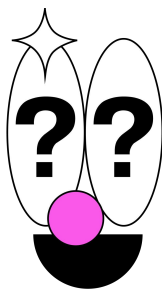


Теперь ваша очередь!

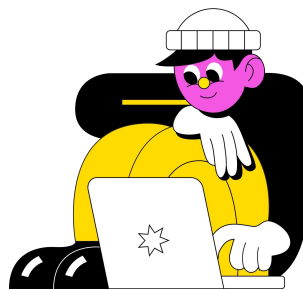
Ответьте на несколько вопросов сообщением в чат



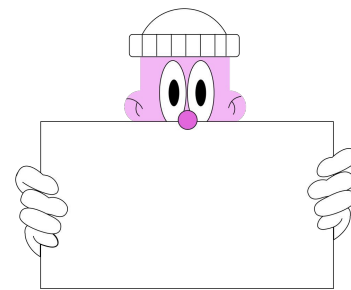
Из какого вы города?



Сколько вам лет?



Кем вы работаете сейчас?



Для чего пришли учиться по направлению "Разработчик"?



Структура семинаров

1

Знакомство с контролем версий Git.
Настройка, основные команды.

2

Работа с ветками в Git.

3

Работа с удалёнными репозиториями в Git - GitHub.



Что будет на уроке сегодня



Quiz!

Ознакомительная интерактивная викторина



Настройка Git и Visual Studio Code



Работа с Git. Составление инструкции по работе с Git

Практическая работа с использованием
языка разметки Markdown

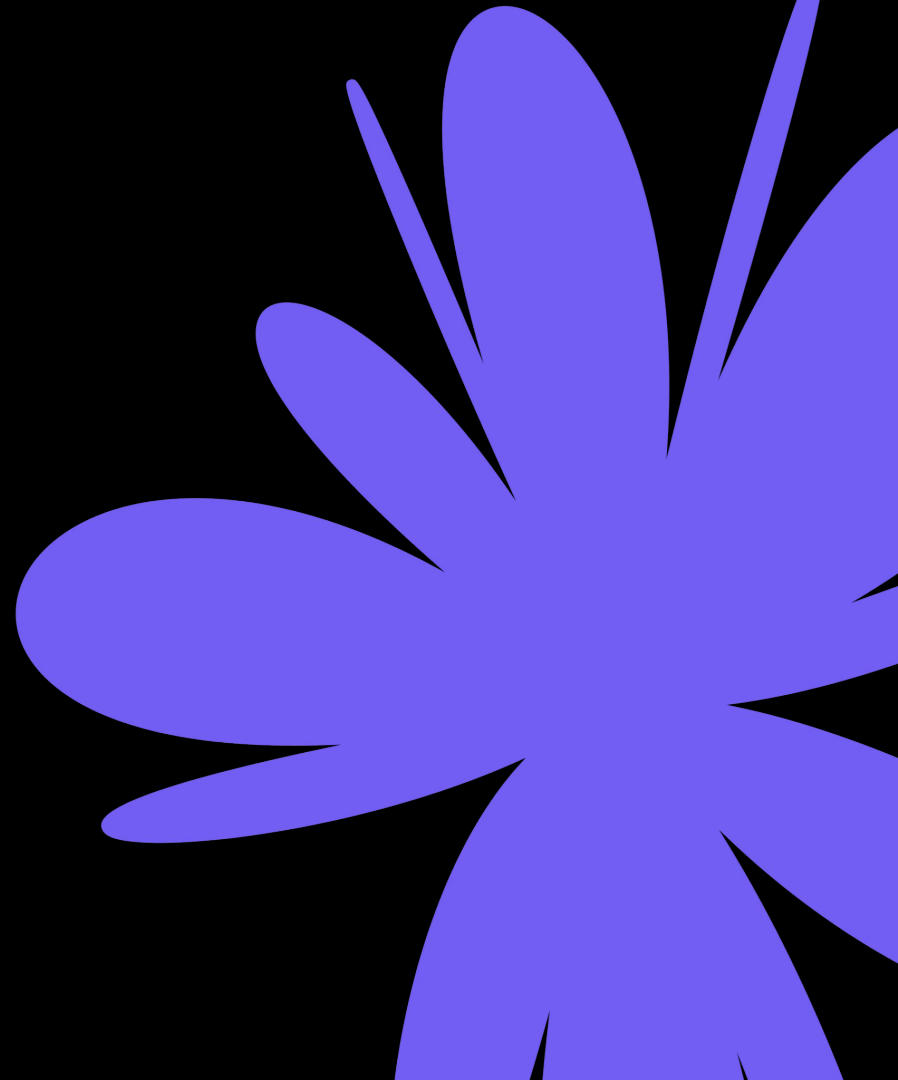


Домашнее задание





Quiz!



С помощью каких символов обрамляется полужирный текст в языке Markdown?

1. ******
2. **-**
3. **##**
4. **<<**



С помощью каких символов обрамляется полужирный текст в языке Markdown?

1. ******
2. -
3. ##
4. <<



Какой способ организации версионности кода предпочтителен в разработке?

1. Организация структуры файлов и папок
2. Использование профессиональных систем контроля версий
3. Использование текстовых онлайн редакторов
4. Пересылка друг другу файлов по почте



Какой способ организации версионности кода предпочтителен в разработке?

1. Организация структуры файлов и папок
2. Использование профессиональных систем контроля версий
3. Использование текстовых онлайн редакторов
4. Пересылка друг другу файлов по почте



Git можно использовать для повседневной работы с текстом.

1. Правда
2. Ложь



Git можно использовать для повседневной работы с текстом.

1. Правда
2. Ложь



Какая команда создает локальный репозиторий?

1. `git init`
2. `git commit`
3. `git push`
4. `git add`



Какая команда создает локальный репозиторий?

1. `git init`
2. `git commit`
3. `git push`
4. `git add`



Репозиторий — это...?

1. Хранилище файлов, поддерживающее версиюность
2. Реализация системы контроля версий
3. Тип базы данных
4. Алгоритм работы с файлами



Репозиторий — это...?

1. Хранилище файлов, поддерживающее версиюность
2. Реализация системы контроля версий
3. Тип базы данных
4. Алгоритм работы с файлами



Какую операцию выполняет команда `git add`?

1. Добавляет файлу версию в локальном репозитории
2. Создаёт локальный репозиторий
3. Отменяет изменения до указанной версии
4. Отправляет файл в удалённый репозиторий



Какую операцию выполняет команда git add?

1. Добавляет файлу версию в локальном репозитории
2. Создаёт локальный репозиторий
3. Отменяет изменения до указанной версии
4. Отправляет файл в удалённый репозиторий



Какая команда фиксирует изменения и сообщает о появлении новых версий файлов?

1. `git log`
2. `git diff`
3. `git commit`
4. `git checkout`



Какая команда фиксирует изменения и сообщает о появлении новых версий файлов?

1. `git log`
2. `git diff`
3. `git commit`
4. `git checkout`



Какая команда показывает разницу между текущей и уже зафиксированной версией файла?

1. `git commit`
2. `git diff`
3. `git commit`
4. `git checkout`



Какая команда показывает разницу между текущей и уже зафиксированной версией файла?

1. `git commit`
2. `git diff`
3. `git commit`
4. `git checkout`



Какая команда выводит список всех коммитов (сохранений) в хронологическом порядке?

1. `git commit`
2. `git diff`
3. `git log`
4. `git checkout`



Какая команда выводит список всех коммитов (сохранений) в хронологическом порядке?

1. `git commit`
2. `git diff`
3. `git log`
4. `git checkout`



Какая команда позволяет перемещаться между сохранениями?

1. `git clone`
2. `git stash`
3. `git log`
4. `git checkout`



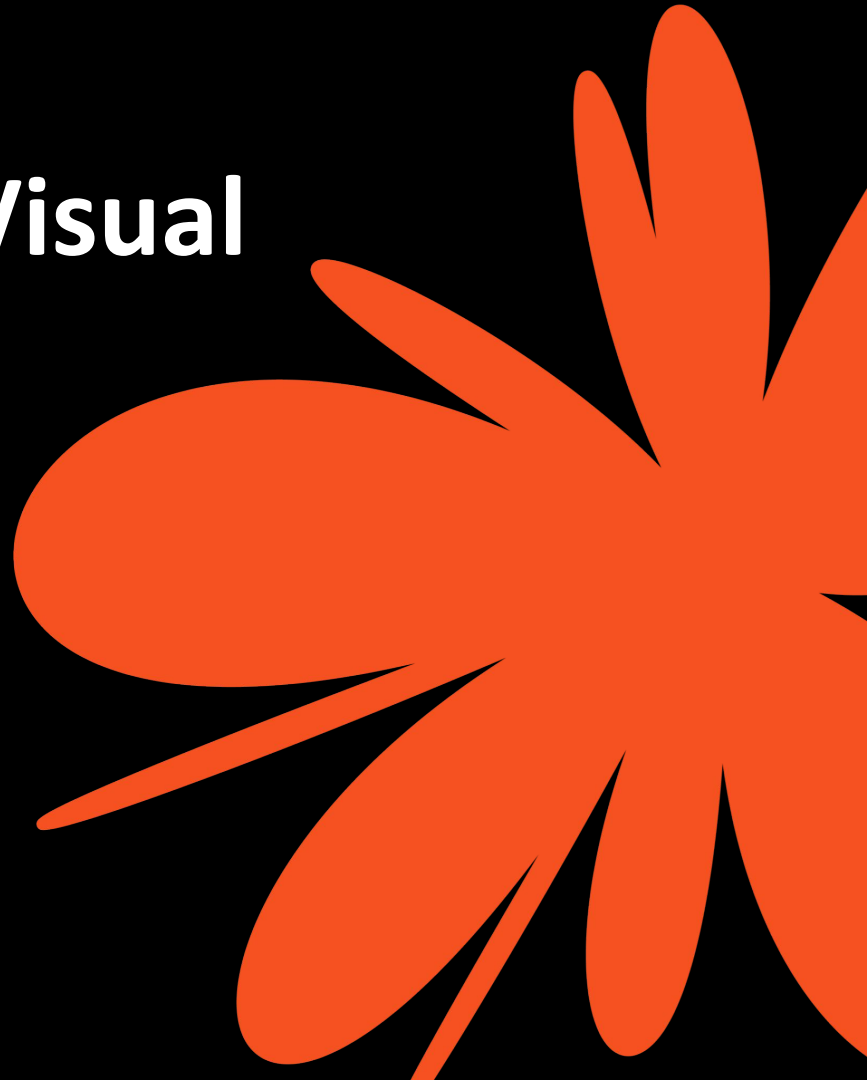
Какая команда позволяет перемещаться между сохранениями?

1. `git clone`
2. `git stash`
3. `git log`
4. `git checkout`





Настройка Git и Visual Studio Code



Установка Git и Visual Studio Code

- Установка Git для Windows, MAC, Linux: <https://git-scm.com/downloads>
- Установка VSCode для Windows, MAC, Linux: <https://code.visualstudio.com/Download>

При первом использовании Git необходимо представиться.

Для этого нужно ввести в терминале 2 команды:

```
git config --global user.name «Ваше имя английскими буквами» git  
config --global user.email ваша почта@example.com
```



Основные команды Git

- ✦ **git init** – инициализация локального репозитория
- ✦ **git status** – получить информацию от git о его текущем состоянии
- ✦ **git add** – добавить файл или файлы к следующему коммиту
- ✦ **git commit -m “message”** – создание коммита.
- ✦ **git log** – вывод на экран истории всех коммитов с их хеш-кодами
- ✦ **git checkout** – переход от одного коммита к другому
- ✦ **git checkout master** – вернуться к актуальному состоянию и продолжить работу
- ✦ **git diff** – увидеть разницу между текущим файлом и закоммиченным файлом



СИНТАКСИС ЯЗЫКА Markdown

Справочник по Markdown от Microsoft:

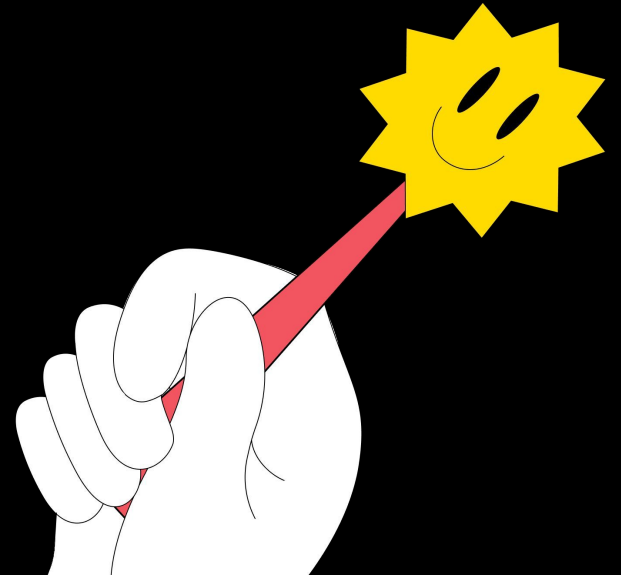
<https://docs.microsoft.com/ru-ru/contribute/markdown-reference>

- ◆ # **Заголовок** – выделение заголовков. Количество символов “#” задаёт уровень заголовка (поддерживается 6 уровней).
- ◆ = **ИЛИ** - – подчёркиванием этими символами (не менее 3 подряд) выделяют заголовки первого (“=”) и второго (“-”) уровней.
- ◆ **** Полужирное начертание**** или **__ Полужирное начертание __**
- ◆ **Курсивное начертание** или *_Курсивное начертание_*
- ◆ *****Полужирное курсивное начертание*****
- ◆ ~~Зачёркнутый текст~~
- ◆ * **Строка** – нумерованные списки, символ “*” в начале строки
- ◆ 1, 2, 3 ... – нумерованные списки





Домашнее задание




Введение в контроль версий. Домашнее задание.

Дооформить инструкцию по работе с Git, используя возможности Markdown (цитаты, картинки, ссылки и др.). Приложить свой проект в заархивированном виде (всю папку целиком).





Спасибо 
за внимание