# Условия домашки по GIT

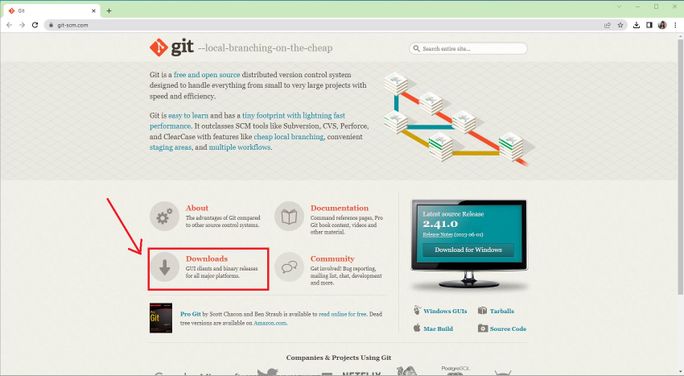
**Задача 1. Создать репозиторий на *GitHub***

Начнем с установки необходимого ПО для работы.

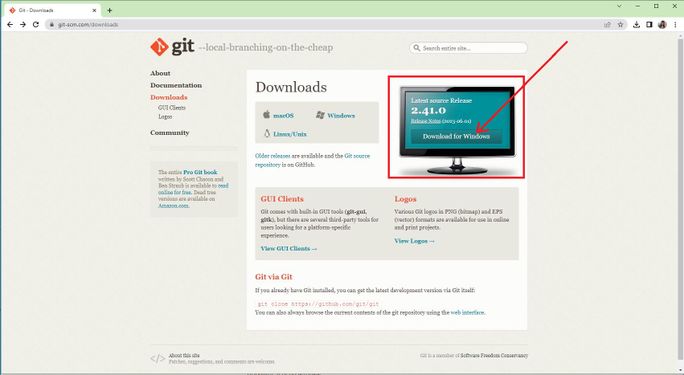
**Шаг 1. Установить Git**

[https://git-scm.com](https://git-scm.com/)  
  
Инструкция для Windows

1. Переходим в раздел Downloads.



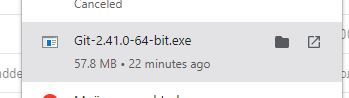
1. Выбираем версию, подходящую для вашей операционной системы.



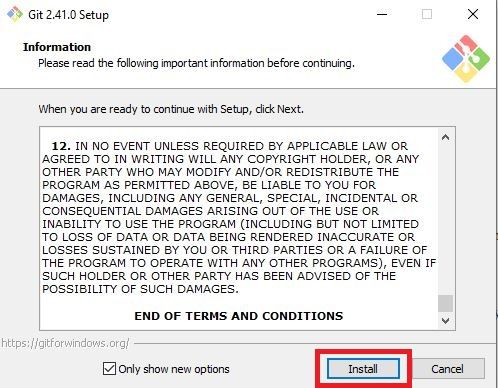
1. Нажимаем Click here to download.

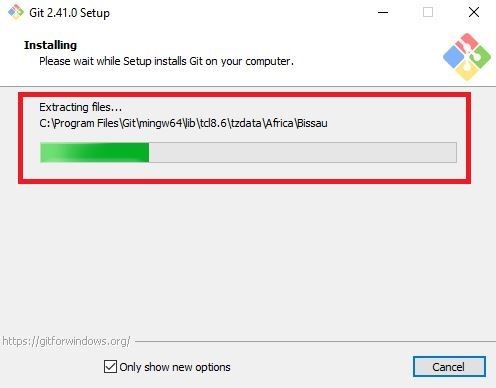


1. Дожидаемся скачивания.



1. Устанавливаем скачанную версию.





**Шаг 2. Зарегистрироваться на GitHub**

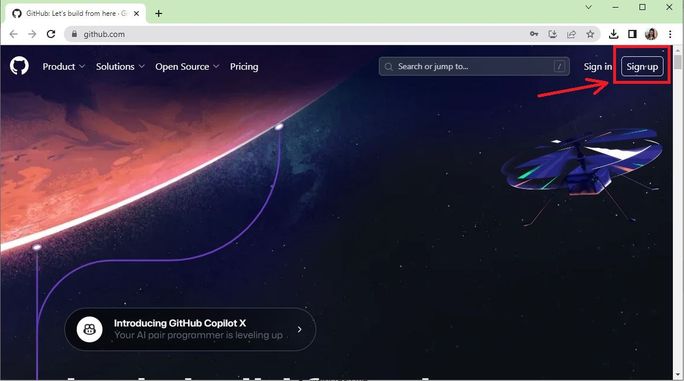
* [https://github.com](https://github.com/)

Как это сделать?

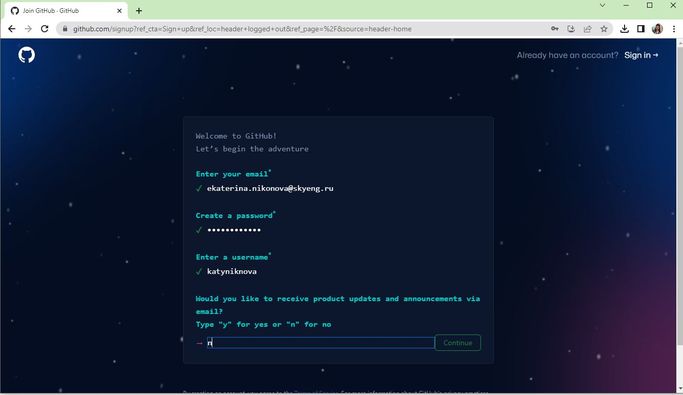
1. Нажимаем

Sign up

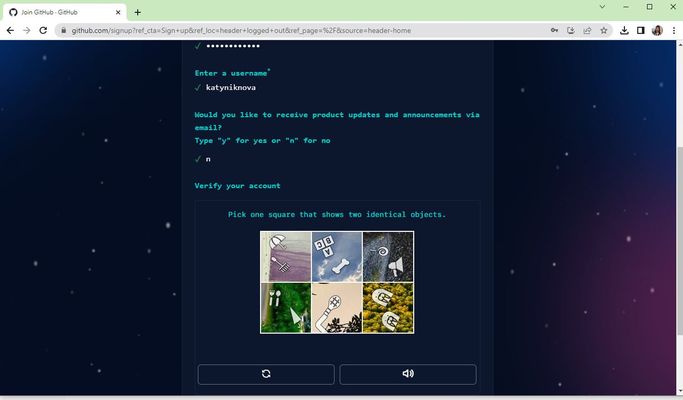
.



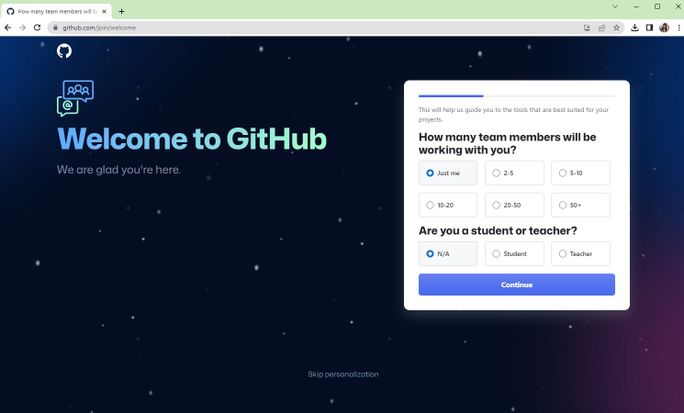
1. Следуем интерактивной инструкции и вводим постепенно email, пароль, имя пользователя и отказ/согласие на подписку.



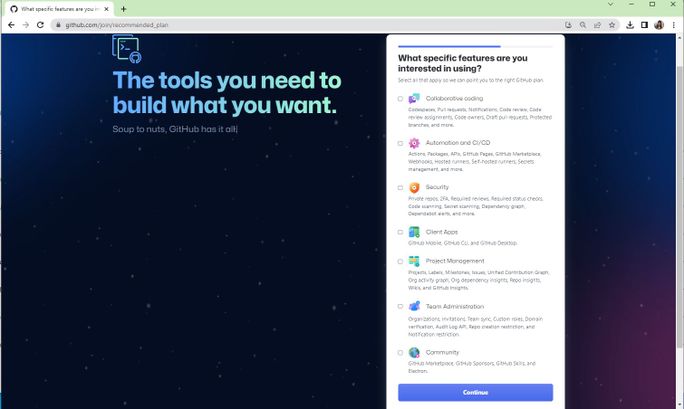
1. Решаем подтверждающй пазл.



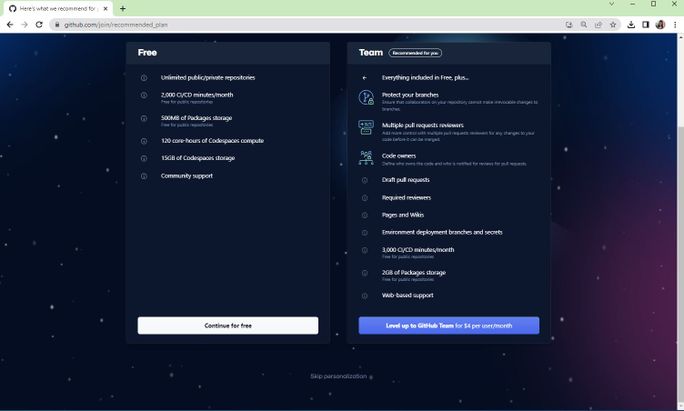
1. Выбираем количество участников, которые будут работать совместно, и статус — «Учитель» или «Студент» (можно пропустить и выбрать N/A).



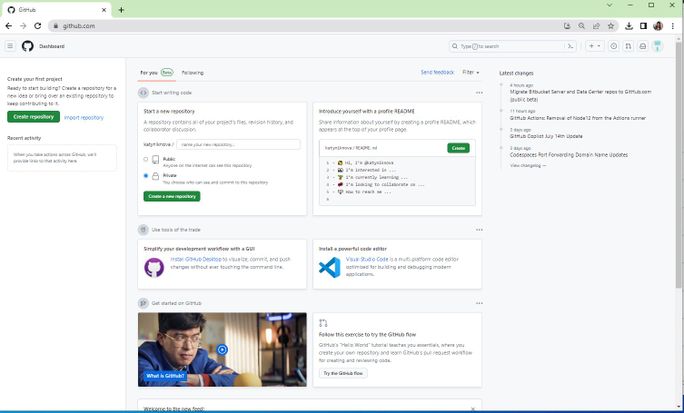
1. При необходимости добавляем дополнения (пока можно пропустить).



1. Выбираем бесплатную версию.



1. Аккаунт создан.

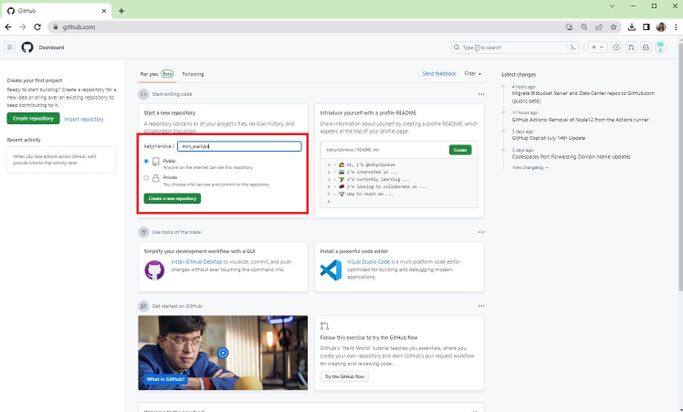


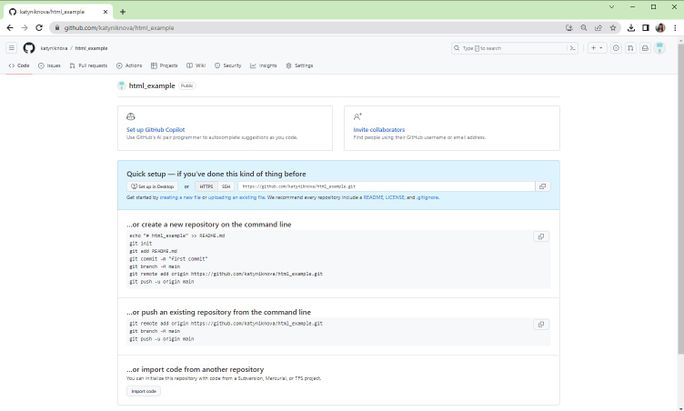
**Шаг 3. Создать репозиторий**

* Создаем репозиторий на GitHub.

Как это сделать?

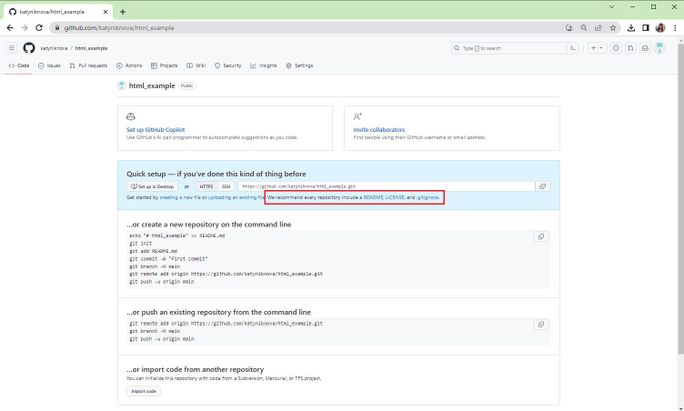
1. При первом создании заполнить название, выбрать Public (для общего доступа).



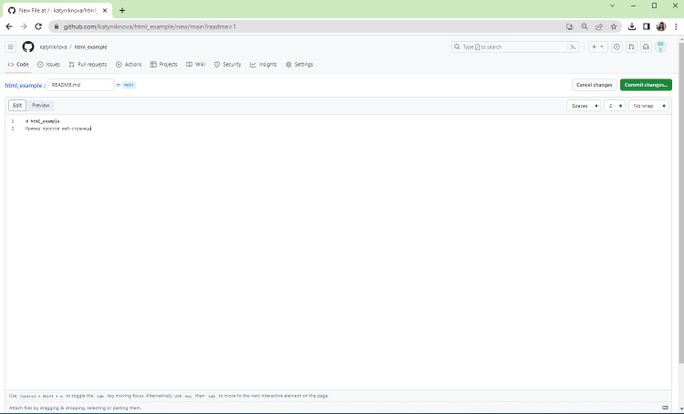


1. Репозиторий создан.

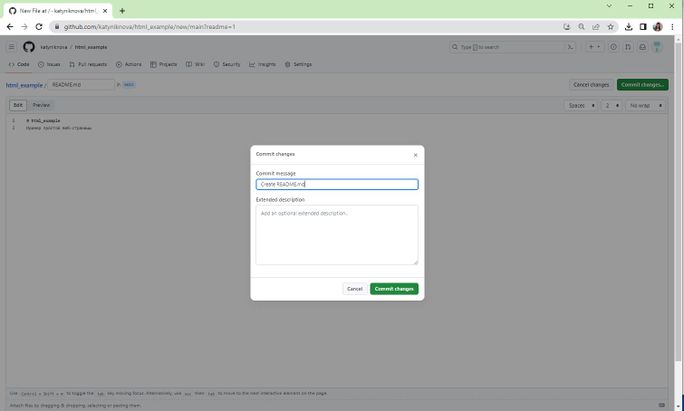
* Создадим основную ветку, для этого добавим и заполним файл README.md.
* В разделе быстрой настройки (Quick setup) выбираем в рекомендациях README.



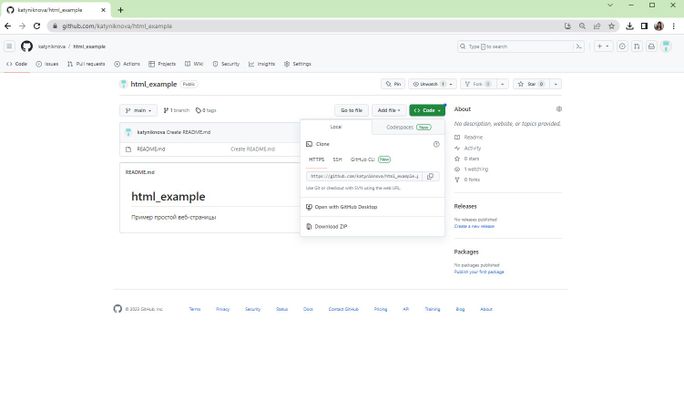
* Заполняем описание будущего проекта.



* Чтобы создать первый коммит, нажимаем на «Commit changes» и заполняем его название.



* Файл успешно добавлен, ветка main создана.



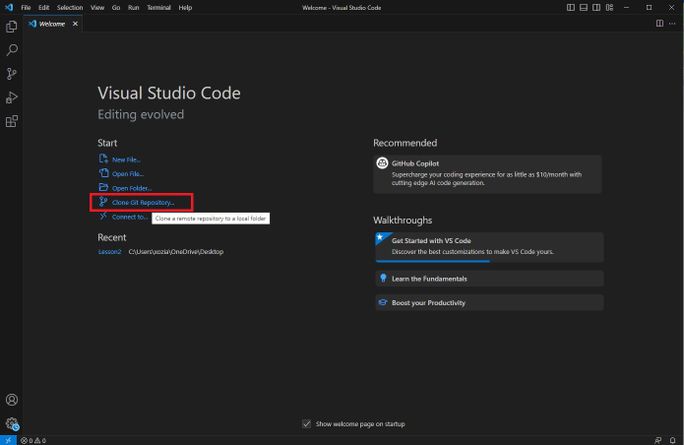
**Шаг 4. Склонировать репозиторий**

Используйте ту IDE, с которой вам удобнее работать.

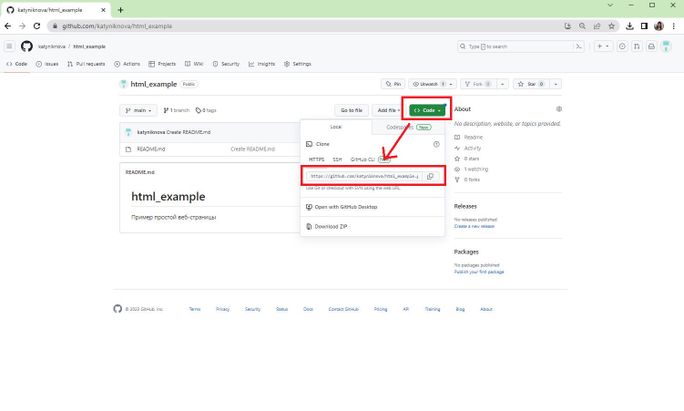
* VSCode: <https://code.visualstudio.com/download>.
* IntelliJ IDEA: <https://www.jetbrains.com/idea/download/>.

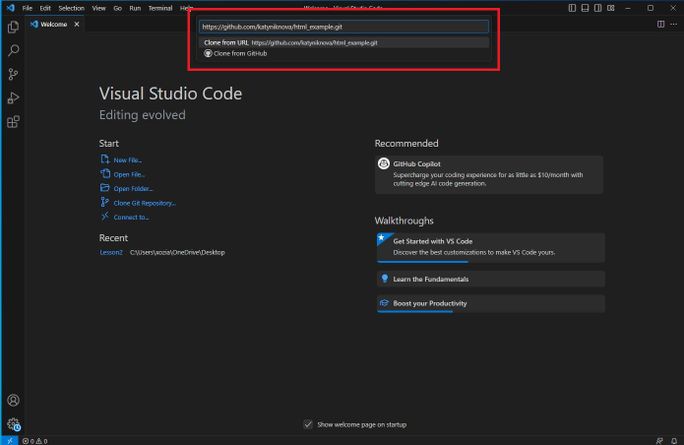
Как это сделать (VSCode)?

1. Выбираем Clone Git Repository.

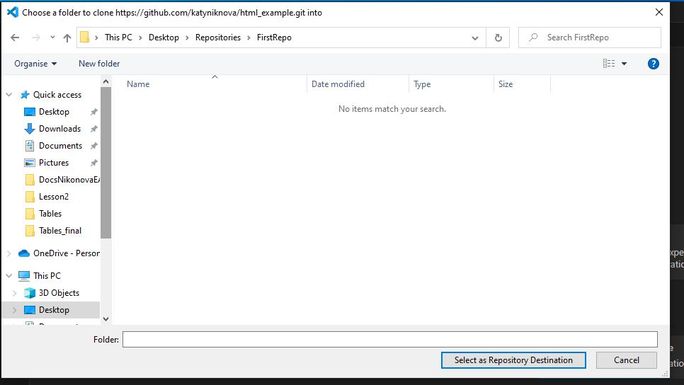


1. Вводим URL созданного на предыдуем шаге репозитория. Для этого нужно предварительно его скопировать.

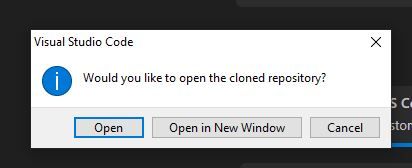


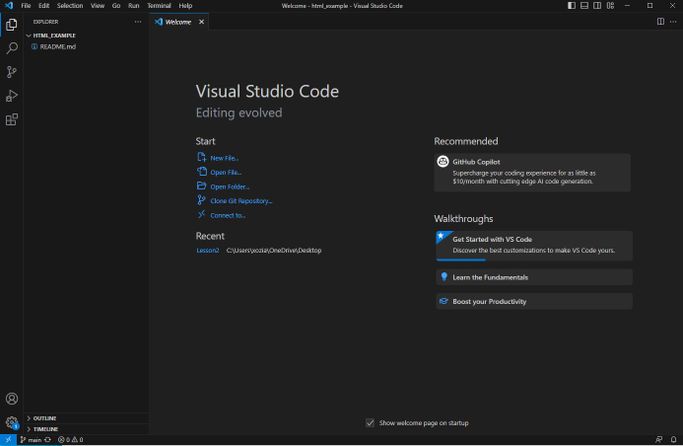


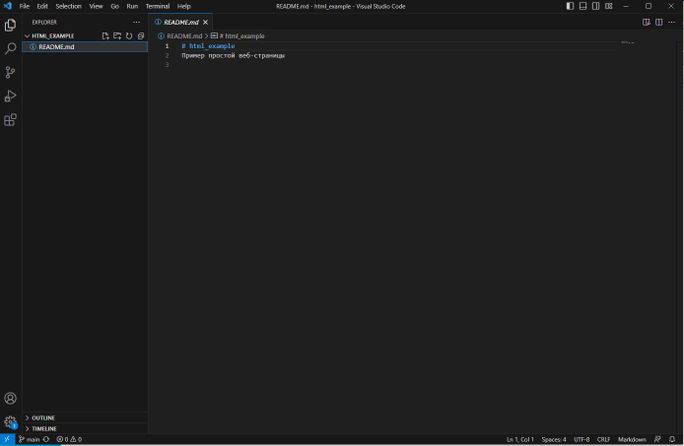
1. Выбираем папку для локальной копии репозитория.



1. Открываем склонированный репозиторий.







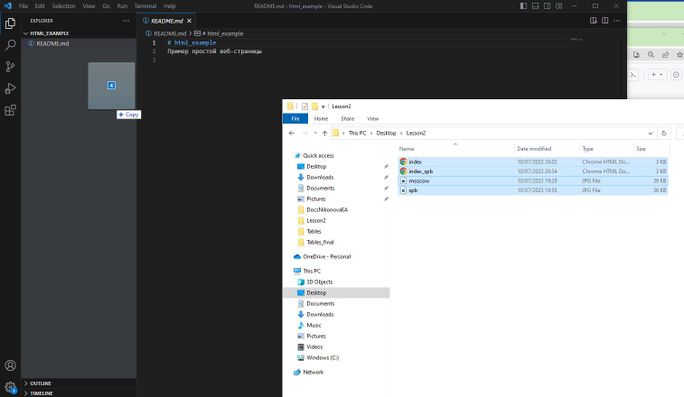
**Шаг 5. Добавьте в репозиторий файлы**

Добавьте в локальный репозиторий файлы из второго домашнего задания.

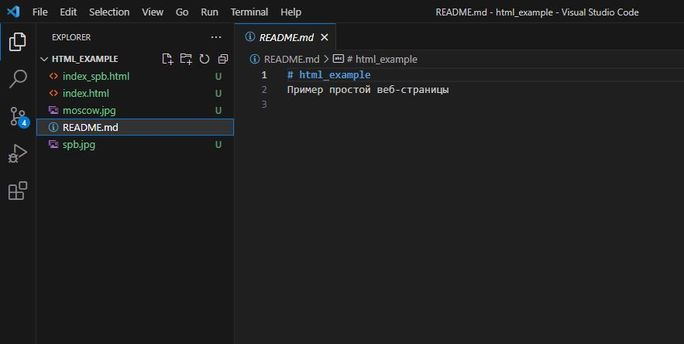
Дополнительно: добавьте дополнительный файлы и/или дополните подробнее файл README. Не забудьте сохранить изменения в файле README.

Как это сделать?

1. Перенесите файлы в пространство проекта, как на скриншоте ниже:



1. Файлы добавлены.



**Шаг 6. Опубликуйте изменения**

Как это сделать?

1. Перейдите в раздел

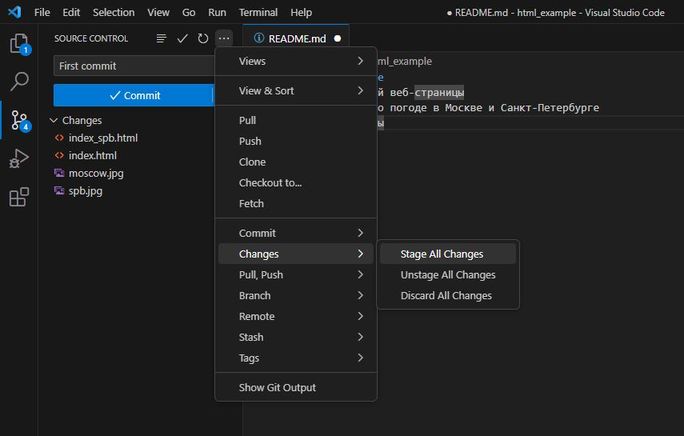
Source Control

 (система управления версиями) на левой панели редактора и заполните сообщение для коммита.

1. Нажмите на три точки и выберите

Changes → Stage All Changes

.

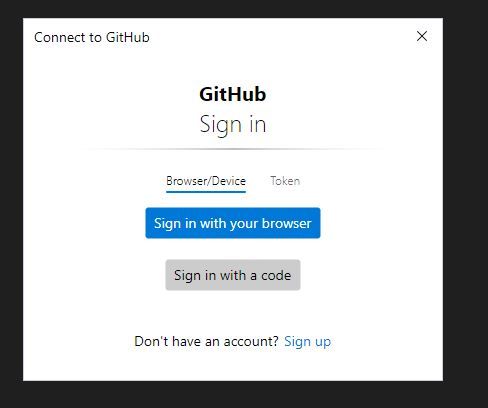


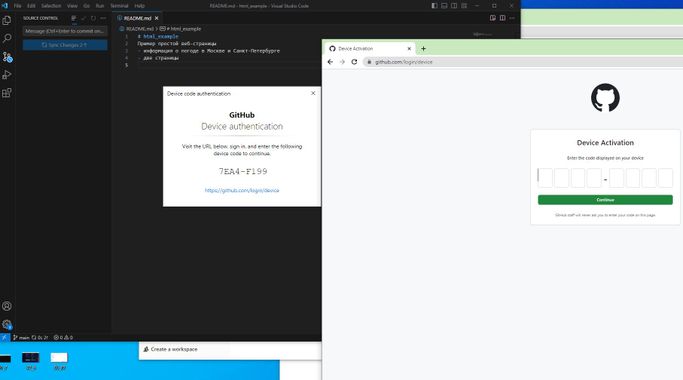
1. Затем нажмите

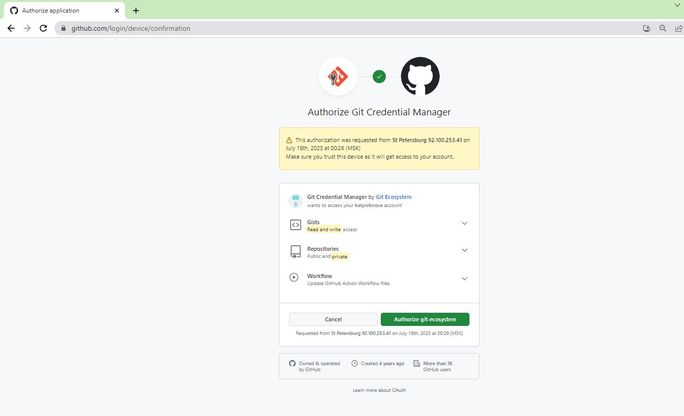
Commit

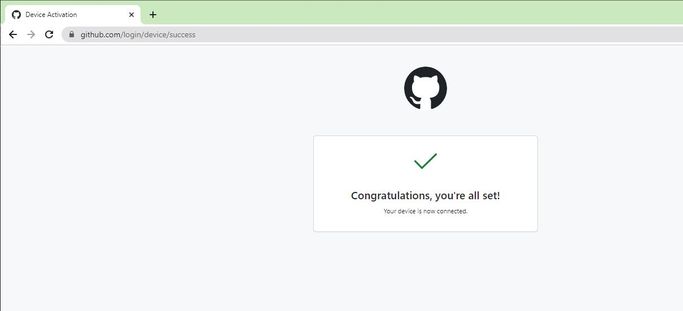
.

1. При необходимости соедините аккаунт с аккаунтом GitHub.









1. Перейдите в репозиторий GitHub и убедитесь, что все изменения добавлены.

**Задание 2. Создать ветки**

Продолжим работу с созданным репозиторием и поработаем больше с терминалом.

**Шаг 1. Создать две ветки**

Создайте от main две ветки —

first\_branch

 и

second\_branch

.

Как это сделать?

1. Откройте терминал в выбранной IDE.
2. Введите команду

git branch first\_branch

.

1. Затем —

git branch second\_branch

.

1. Для проверки создания веток выполните команду

git branch -a

. Команда выводит все ветки — и локальные, и удаленные. Для вывода только локальных веток используйте

git branch

. Соответственно, для вывода только удаленных веток выполните

git branch -r

. Рекомендовано выполнить

git fetch

 перед проверкой удаленных веток для получения последних изменений из удаленного репозитория.



**Шаг 2. Добавить два разных файла в каждую ветку**

* Переключитесь на ветку

first\_branch

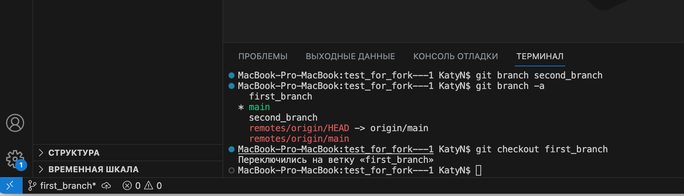
 и добавьте, например, текстовый файл. Опубликуйте ветку и изменения.

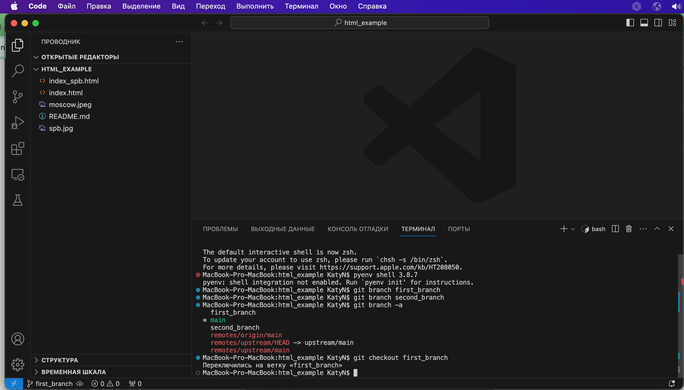
Как это сделать?

* 1. Для того чтобы переключиться на ветку, используйте команду

git checkout first\_branch

.





* 1. Затем добавьте, например, текстовый файл test.txt в файлы проекта. Можно вручную, можно в терминале командой

echo Some text>test.txt.

.

* 1. Выполните команду

git add test.txt

 для фиксации изменений.

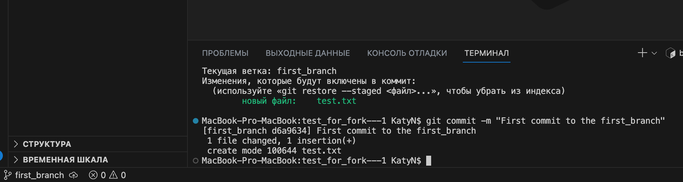
* 1. Выполните команду

git status

 для проверки включения изменений в будущий коммит.

* 1. Выполните коммит изменений командой

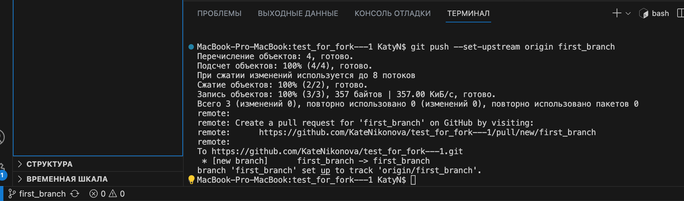
git commit -m "First commit to the first\_branch"



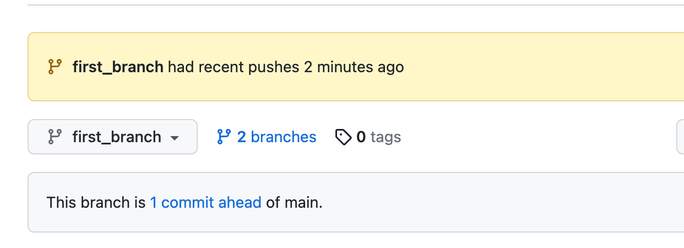
* 1. Отправьте изменения на удаленный репозиторий, выполнив команду

git push --set-upstream origin first\_branch

. Так как ветка еще не существовала в удаленном репозитории, необходимо выполнить именно такую команду.



* 1. Перейдите в GitHub и убедитесь, что ветка и ее изменения там появились.



* Переключитесь на ветку

second\_branch

 и добавьте, например, картинку. Опубликуйте ветку и изменения аналогично изменениям в первую ветку.

**Задание 3. Сделать pull request**

Продолжим работу с созданным репозиторием и создадим pull request.

**Pull request** — запрос на внесение изменений. Когда разработчик завершает работу над своими изменениями в отдельной ветке (branch), он создает pull request, чтобы запросить вливание этих изменений в главную ветку проекта. Другие члены команды могут просмотреть изменения, оставить комментарии, протестировать и утвердить pull request, прежде чем он будет объединен с основным кодом. Этот процесс помогает поддерживать качество кода и обеспечивает прозрачность и понимание изменений в проекте.

* Перейдите в удаленный репозиторий.
* Нажмите на кнопку Pull requests в верхней части страницы.
* Нажмите на зеленую кнопку New pull request.
* Выберите ветку, в которой содержатся ваши изменения, —

first\_branch

, и ветку, в которую вы хотите внести эти изменения, —

main

.

* Нажмите на кнопку Create pull request.
* Измените название по шаблону First\_PR\_Ivanov\_95.2, где Ivanov — ваша фамилия латиницей, а 95.2 — номер вашей группы.
* Нажмите на кнопку Create pull request для завершения создания pull request.
* Скопируйте ссылку из адресной строки браузера.

Формат сдачи основного задания: ссылка на pull request.

Например, <https://github.com/skyproshik/sky_examples/pull/1>.

## ДОПОЛНЕНИЯ Git. Локальный репозиторий

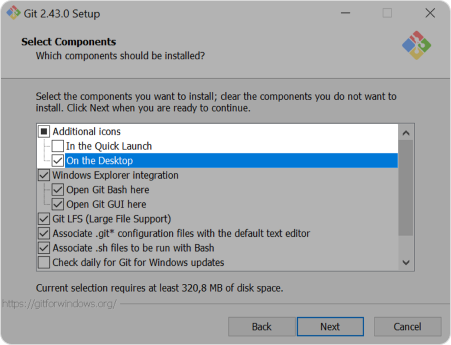
**Git** — распределенная система контроля версий.

**Git-репозиторий** — папка, в которой хранится код проекта и все его версии.

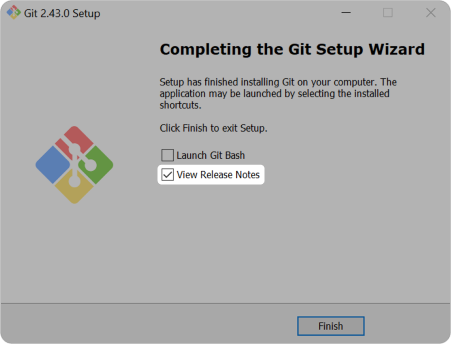
### Как установить и настроить Git локально

1. Перейдите на страницу установки Git: <https://git-scm.com/downloads>.
2. Скачайте дистрибутив, который подходит вашей операционной системе.
3. Перейдите в папку «Загрузки» и запустите установочный файл. На экране отобразятся параметры установки — везде нажимайте Next («Далее»).

*Если вы хотите создать иконку Git на рабочем столе, выберите Additional icons → On the Desktop.*



*Если вы хотите получать уведомления об обновлениях Git, выберите View Release Notes.*



1. Когда установка завершится, откройте командную строку и введите команду

git -v

. Эта команда выведет версию Git, если он правильно установлен.

Мы установили Git. Это значит, что теперь с ним можно работать локально.

### Как создать локальный репозиторий

**Локальный репозиторий** — это хранилище данных на компьютере пользователя.

Локальным репозиторием может быть папка на рабочем столе. В ней можно хранить все документы проекта, а в отдельном файле папки — историю изменения репозитория. **Создадим локальный репозиторий с помощью Git.**

1. Запустим командную строку.
2. Перейдем на рабочий стол:

cd Desktop

.

1. Выведем список папок на рабочем столе:

dir

.

1. Найдем или создадим пустую папку. Назовем ее first\_repo — эта папка станет нашим репозиторием.
2. Перейдем к папке first\_repo:

cd first\_repo

.

1. Инициализируем репозиторий:

git init

.

**Инициализация (initialisation)** репозитория — создание репозитория.

1. Проверим наличие скрытых файлов в репозитории. Введем команду

dir /ah

. Появится скрытый файл .git. В него будут сохраняться все изменения в репозитории.

## GitHub. Удаленный репозиторий

**Удаленный репозиторий** — репозиторий на удаленном сервере, в котором хранятся версии кода и файлы проекта.

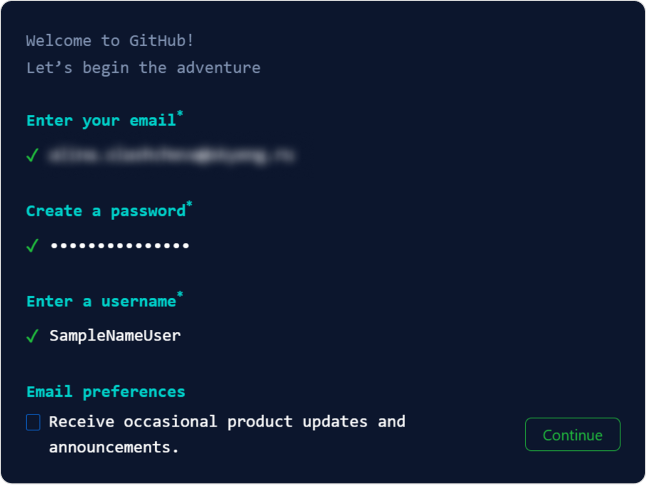
Мы зарегистрируемся и поработаем с удаленным репозиторием в GitHub. **GitHub** — это онлайн-сервис, в котором разработчики вместе создают и поддерживают проект и делятся своим кодом. Чтобы создать удаленный репозиторий, зарегистрируемся в GitHub.

**Регистрация в GitHub**

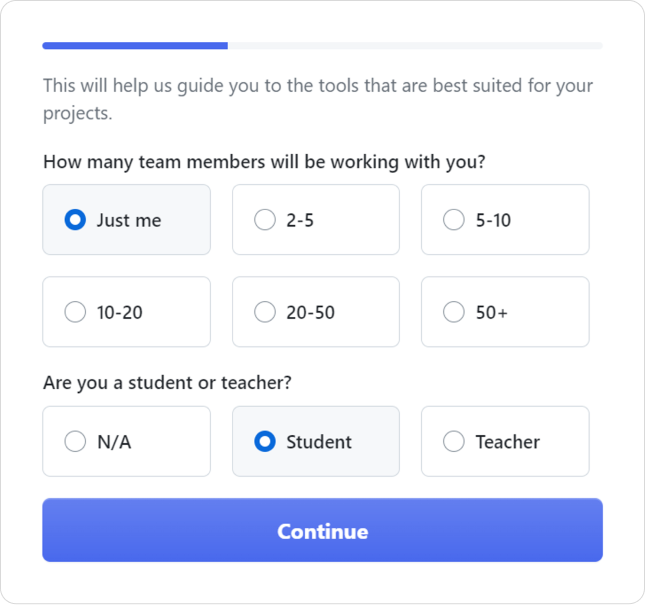
1. Перейдем на страницу регистрации и введем свои данные.

*Receive occasional product updates and announcements — выберите этот пункт, если хотите получать уведомления об обновлениях GitHub.*

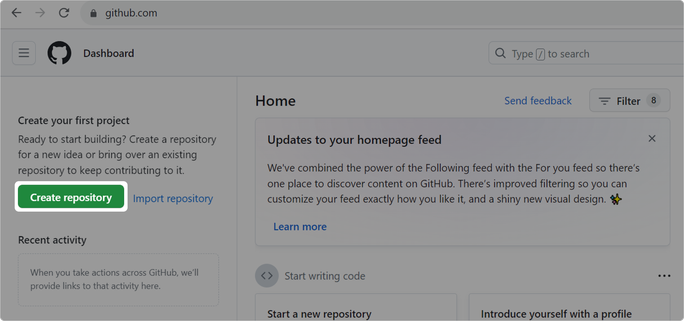
Нажмем Continue.



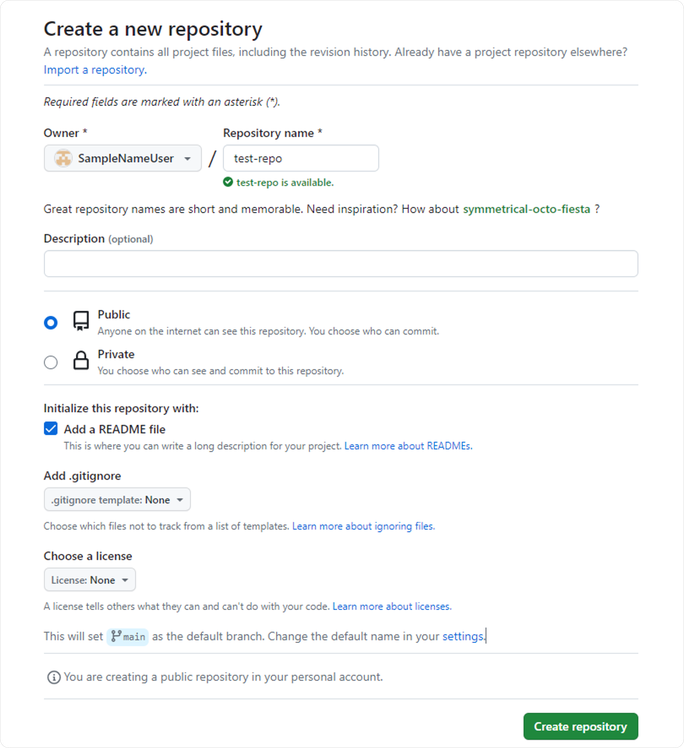
1. Пройдем верификацию.
2. Подтвердим учетную запись, отправив письмо с кодом на электронную почту. Затем скопируем код и вставим в GitHub.
3. Введем информацию о себе:



1. Перейдем в раздел создания репозитория.



1. Назовем свой репозиторий и добавим информацию о нём. Создадим репозиторий, нажав на кнопку Create repository.

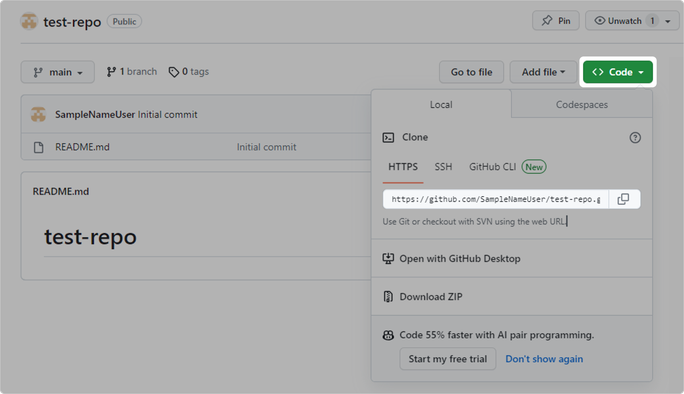


Мы создали публичный удаленный репозиторий, которым можем делиться с другими пользователями. Название нашего репозитория — test-repo. В нём хранится один файл — README.

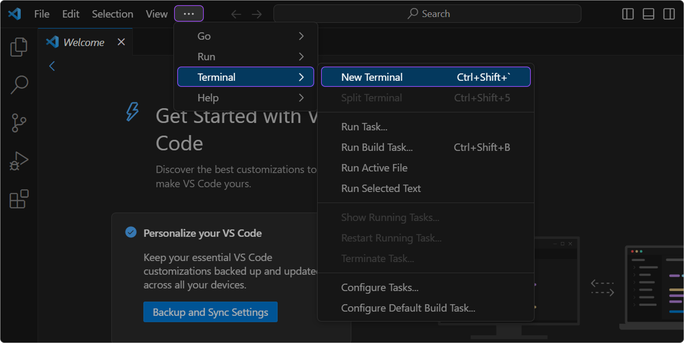
## Изменения на удаленном и локальном репозитории

Мы создали локальный и удаленный репозитории. Чтобы вносить изменения в локальном репозитории, затем отправлять их на удаленный и наоборот, **репозитории нужно объединить**.

1. Скопируем ссылку на репозиторий.



1. Откроем терминал в VS Code.



1. Перейдем на рабочий стол:

cd Desktop

.

1. Склонируем удаленный репозиторий с GitHub себе на компьютер с помощью команды:

git clone ссылка с GitHub

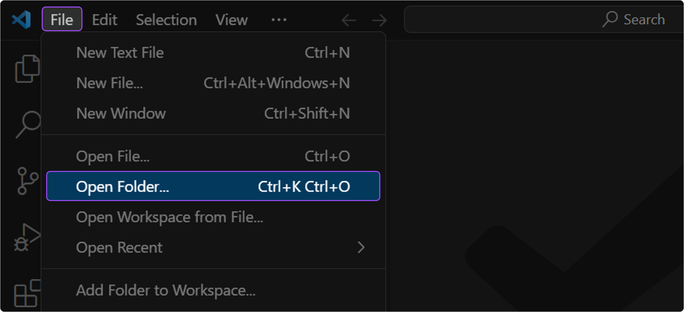
.

На рабочем столе появится папка test-repo — копия нашего удаленного репозитория. В ней есть файл README.

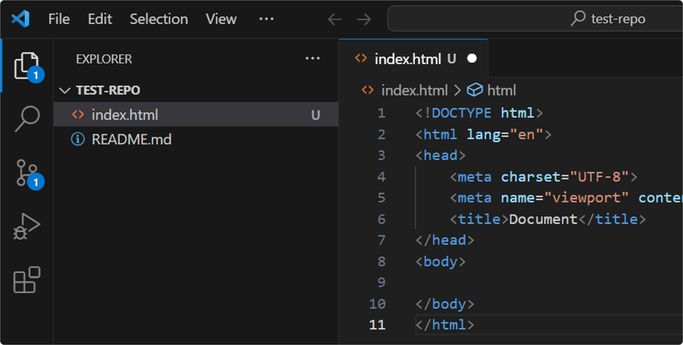
## Локальная работа с удаленным репозиторием

Добавим локальный файл в **копию удаленного репозитория**.

1. Откроем папку test-repo в VS Code.



1. Добавим в репозиторий локальный файл index.html и в нём создадим шаблон HTML-страницы.



Теперь в локальном репозитории два файла: index.html и README. Удаленный репозиторий на GitHub хранит только файл README. **Добавим файл index.html из локального репозитория в удаленный.**

1. Введем команду

git status

 в терминал VS Code. Так мы посмотрим, какие изменения произошли в test-repo. Видим, что Git не отслеживает файл index.html.

1. Чтобы добавить его в отслеживаемые файлы, введем команду

git add index.html

.

1. Проверим статус файла командой

git status

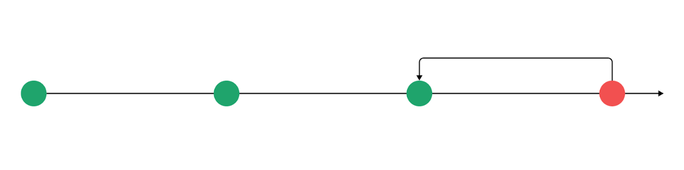
. Предыдущая команда выполнена успешно, поэтому файл index.html стал отслеживаемым.

1. Введем команду

git commit -m "first commit"

 — **сделаем коммит**. Так мы покажем Git, что файл изменен. Изменения файла сохранятся в репозитории.

**Коммит (commit)** — операция записи изменений в репозиторий Git.



Sources:

*Горизонтальная линия — процесс разработки продукта. Кружочки — коммиты. Они как метки — показывают, что версия предыдущего файла была изменена.*

1. Внесем изменения на удаленный репозиторий командой

git push

.

Ошибки

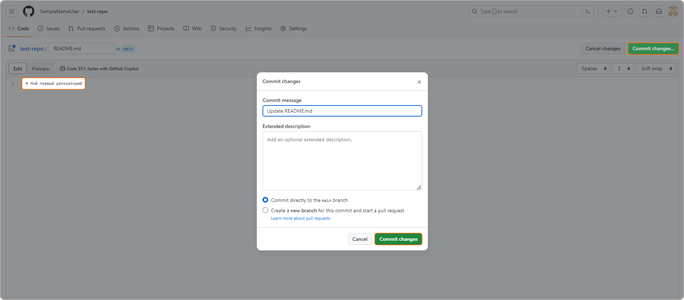
Изменения внесены на удаленный репозиторий на GitHub, там появился файл index.html. Проверим изменения через терминал: введем команду

git log

. Она выведет список всех наших коммитов.

**Внесем изменения на удаленном репозитории и отправим их на локальный репозиторий.**

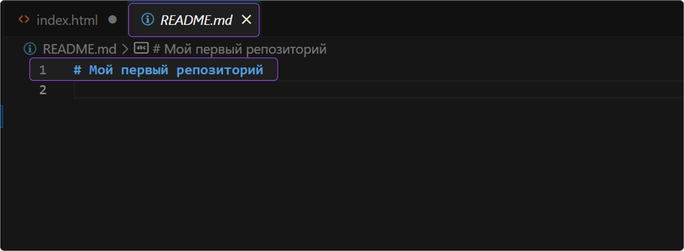
1. Перейдем в файл README в удаленном репозитории GitHub.
2. Введем «Мой первый репозиторий».
3. Нажмем Commit changes.



1. В терминале VS Code введем команду

git pull

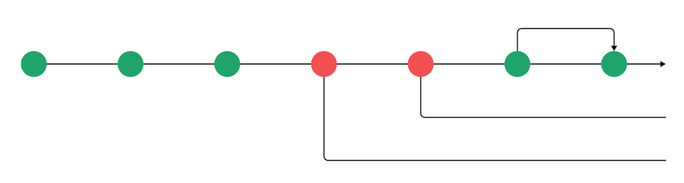
 — она подтянет изменения из удаленного репозитория в локальный.



## Ветки проекта

**Ветка (branch)** — отдельная линия разработки, которая отклоняется от основной ветки.

Основную ветку называют мастер-веткой или главной веткой. Она содержит файлы и код рабочего проекта. Если проект нужно доработать, например написать новый код, создают копию главной ветки и уже в нее вносят изменения. Эти изменения не влияют на работу проекта, пока их не занесут в главную ветку.



Sources:

Горизонтальная линия — главная ветка. Кружочки — коммиты в рабочем проекте. Красные кружочки — коммиты, на которых создается отдельная ветка. Она наследует историю коммитов главной ветки до момента своего создания.

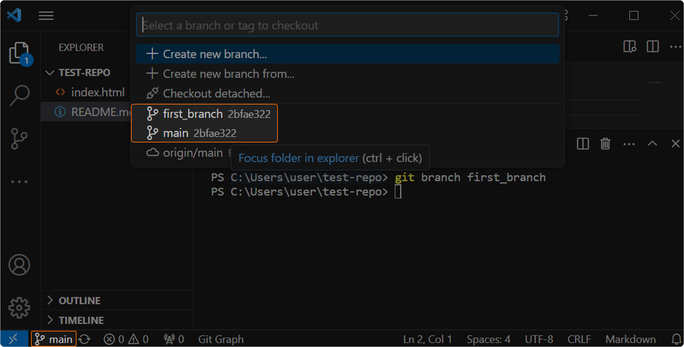
Создадим первую ветку в локальном репозитории. Введем команду

git branch

 и название новой ветки — first\_branch:

git branch first\_branch

Нажмем на значок ветки в левом нижнем углу VS Code, чтобы посмотреть список веток. Рядом со значком указано название ветки, в которой мы сейчас работаем.



В нашем локальном репозитории теперь две ветки: main — главная, first\_branch — копия главной ветки. Чтобы переключиться между ветками, используем команду

git checkout

и название ветки, на которую хотим перейти:

git checkout first\_branch

Добавим в ветку файл index2.html. Сделаем его отслеживаемым:

git add index2.html

Посмотрим статус проекта:

git status

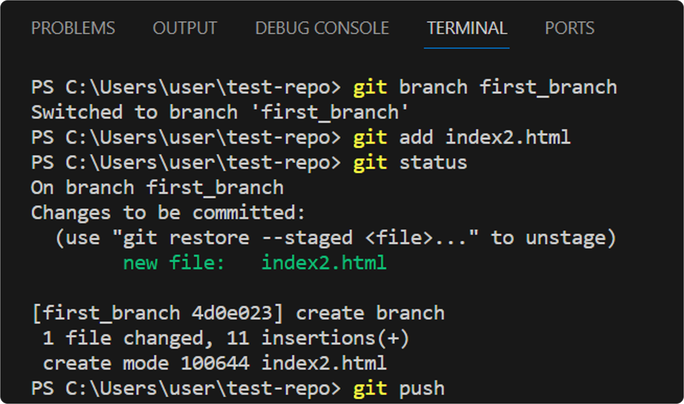
Git выделил название файла зеленым. Это значит, что он следит за изменениями файла.

Теперь закоммитим изменения, чтобы они сохранились в локальном репозитории:

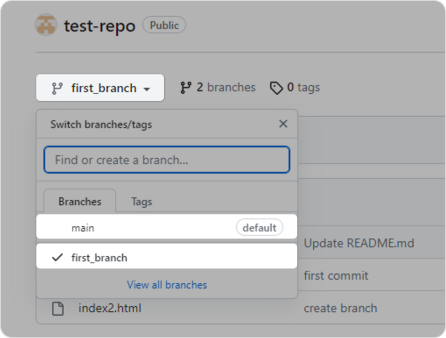
git commit -m "create branch"

И запушим их в удаленный репозиторий:

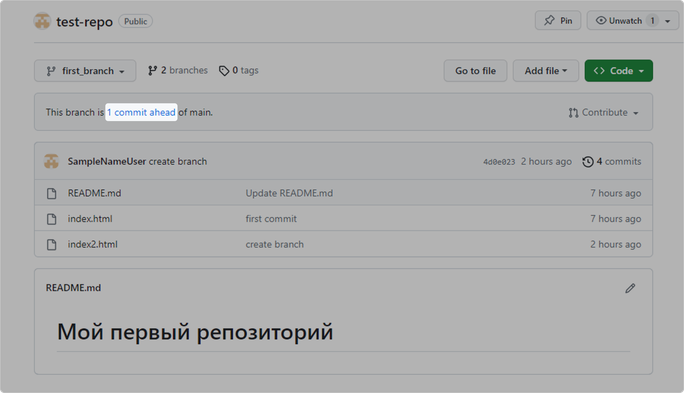
git push



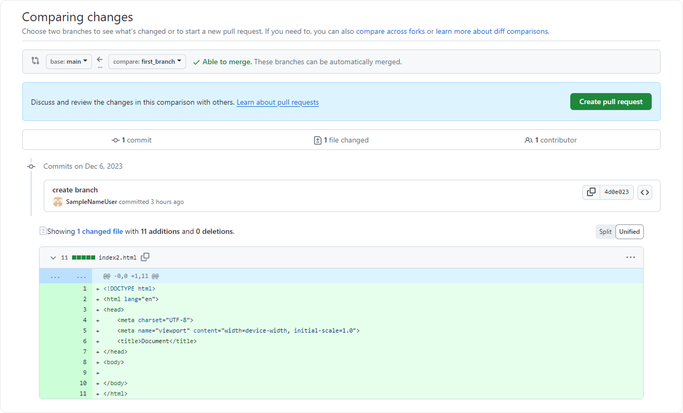
Все изменения запушились в удаленный репозиторий. Перейдем в него и сравним наши ветки: main и first\_branch. Мы можем переключаться между ветками и смотреть их содержимое:



Перейдем к ветке first\_branch. GitHub подсказывает, что она была создана от главной ветки в результате одного коммита. Нажимаем на это сообщение — ссылка в нем перенесет нас в раздел **Comparing changes**.



Здесь мы посмотрим, какие изменения мы вносили в ветку и чем она отличается от главной.



Ветка first\_branch отличается от ветки main тем, что содержит файл index2.html.

## Команды Git

git init

 — создание нового репозитория Git в пустой директории или превращение существующей директории в новый репозиторий Git.

git clone <URL>

 — клонирование репозитория из удаленного хранилища на локальную машину.

git push

 — выгрузка сделанных изменений в удаленный репозиторий (например, GitHub).

git pull

 — получение изменений из удаленного репозитория на локальную машину.

git status

 — просмотр списка измененных файлов.

git branch название ветки

 — создание ветки.

git checkout название ветки` — переход на ветку.

git commit

 — сохранение изменений в Git.

git log

 — просмотр истории коммитов в репозитории.

Тестировщики могут использовать СКВ, чтобы просматривать код и сравнивать изменения в разных версиях, самостоятельно разворачивать необходимую версию приложения и/или определенного сервиса на нужную тестовую среду. А автоматизаторы тестирования работают с кодом и используют системы контроля версий.

# Практика

10 из 11

Выполните упражнения тренажера: <https://git-sky.herokuapp.com/?locale=ru_RU>.

