

## Практическое руководство 1. Применение GitHub

В этом руководстве собраны рекомендации по изучению возможностей применения GitHub и Git для решения своих задач. Источник – официальная документация <https://docs.github.com/ru>.

### *Сведения о GitHub и Git*

<https://docs.github.com/ru/get-started/start-your-journey/about-github-and-git>

Для совместной работы можно использовать GitHub и Git.

#### О GitHub

GitHub — это облачная платформа, на которой вы можете хранить код, делиться им и работать вместе с другими над написанием кода.

Хранение вашего кода в «репозитории» на GitHub позволяет вам:

- **Продемонстрировать или поделиться** своей работой.
- **Отслеживать и управлять** изменениями в вашем коде с течением времени.
- Позволить другим **просматривать** ваш код и вносить предложения по его улучшению.
- **Работать совместно** над общим проектом, не беспокоясь о том, что ваши изменения повлияют на работу ваших соавторов, прежде чем вы будете готовы их интегрировать.

Совместная работа, одна из фундаментальных функций GitHub, стала возможной благодаря программному обеспечению с открытым исходным кодом Git, на котором построен GitHub.

#### О Git

Git — это система контроля версий, которая интеллектуально отслеживает изменения в файлах. Git особенно полезен, когда вы и группа людей одновременно вносите изменения в одни и те же файлы.

Обычно, чтобы сделать это в рабочем процессе на основе Git, вам нужно:

- **Создать ответвление (branch – ветка)** от основной копии файлов, над которыми вы (и ваши коллеги) работаете.
- **Внести изменения** в файлы самостоятельно и безопасно в своей личной ветке.
- Использовать Git для разумного **объединения** ваших изменений обратно в основную копию файлов, чтобы ваши изменения не влияли на обновления других людей.
- Позволить Git **отслеживать** ваши изменения и изменения других людей, чтобы вы все продолжали работать над самой последней версией проекта.

## Как Git и GitHub работают вместе?

Когда вы загружаете файлы на GitHub, вы сохраняете их в «репозитории Git». Это означает, что, когда вы вносите изменения (или «фиксируете») в свои файлы на GitHub, Git автоматически начинает отслеживать ваши изменения и управлять ими.

Существует множество действий, связанных с Git, которые вы можете выполнить на GitHub прямо в браузере, например создание репозитория Git, создание ветвей, а также загрузка и редактирование файлов.

Однако большинство людей работают над своими файлами локально (на своем компьютере), а затем постоянно синхронизируют эти локальные изменения — и все связанные с ними данные Git — с центральным удаленным (remote) репозиторием на GitHub. Для этого можно использовать множество инструментов, например **GitHub Desktop**.

Как только вы начнете сотрудничать с другими людьми и всем вам придется работать над одним и тем же репозиторием одновременно, вы будете постоянно:

- **Получать (Pull)** все последние изменения, внесенные вашими соавторами, из удаленного репозитория на GitHub.
- **Отправлять (Push)** свои изменения в тот же удаленный репозиторий на GitHub.

Git выясняет, как разумно объединить этот поток изменений, а GitHub помогает вам управлять этим потоком с помощью таких функций, как «запросы на извлечение».

## С чего мне начать?

Если вы новичок в GitHub и не знакомы с Git, рекомендуем ознакомиться со статьями в категории «[Начните свое путешествие](#)». В этих статьях основное внимание уделяется задачам, которые вы можете выполнить прямо в браузере на GitHub и они помогут вам:

- **Создать учетную запись** на GitHub.
- **Изучить «GitHub Flow»** и ключевые принципы совместной работы (ветвления, фиксации, запросы на включение, слияния).
- **Настроить (персонализировать) свой профиль**, чтобы поделиться своими интересами и навыками.
- **Исследовать GitHub**, чтобы найти вдохновение для своих собственных проектов и пообщаться с другими.
- Узнать, как **загрузить** код, который вас заинтересовал для собственного использования.
- Узнать, как **загрузить** то, над чем вы работаете в репозиторий GitHub.

## Создание учетной записи на GitHub

<https://docs.github.com/ru/get-started/start-your-journey/creating-an-account-on-github>

Описывается как создается личная учетная запись для начала работы с GitHub.

### Сведения о personal account на GitHub.com

Чтобы приступить к работе с GitHub, необходимо создать **бесплатную** личную учетную запись на GitHub.com и проверить адрес электронной почты.

Каждый пользователь, использующий GitHub.com, входит в личную учетную запись. Ваш личная учетная запись — это ваше удостоверение на GitHub.com, включающее имя пользователя и профиль.

### Регистрация нового личная учетная запись

1. Перейдите к <https://github.com/>.
2. Щелкните **Sing Up (Регистрация)**.
3. Следуйте инструкциям, чтобы создать личная учетная запись.

Во время регистрации вам будет предложено проверить адрес электронной почты. Без проверенного адреса электронной почты вы не сможете выполнить некоторые основные задачи GitHub, например создание репозитория.

Если у вас возникли проблемы с проверкой адреса электронной почты, можно выполнить некоторые действия по устранению неполадок. Дополнительные сведения см. в разделе [Подтверждение адреса электронной почты](#).

Теперь, когда вы создали личная учетная запись, вы сможете изучить основы GitHub.

В руководстве "[Hello World](#)" вы сможете узнать о репозиториях и их способах создания, а также о таких понятиях, как ветвление, фиксации и запросы на вытягивание.

### Упражнение 1. Базовые навыки - Hello World

В этом упражнении вы познакомитесь с основными понятиями GitHub, такими как репозитории, ветки, коммиты, запросы на включение (repositories, branches, commits, pull requests). Вы создадите свой собственный репозиторий **Hello World** и изучите процесс запросов на включение (pull request) GitHub — популярный способ создания и проверки кода.

В этом кратком руководстве вы:

- Создадите и примените репозиторий.
- Запустите новую ветку.
- Внесение изменения в файл и отправка их на GitHub как коммиты.
- Создадите и выполните запрос на включение.

## Замечание

- Для выполнения данного упражнения у вас должна быть учетная запись GitHub и вам не нужно знать, как программировать, использовать командную строку или устанавливать Git (программное обеспечение для контроля версий, на котором построен GitHub).

## Шаг 1. Создание репозитория

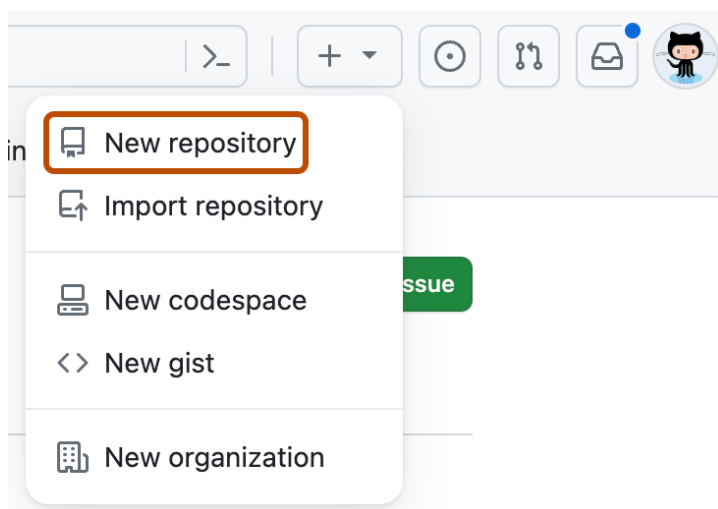
Первое, что нужно сделать – создать репозиторий. Можно представить репозиторий как папку, содержащую связанные элементы, такие как файлы, изображения, видео или даже другие папки. В репозитории обычно группируются элементы, принадлежащие одному и тому же «проекту» или объекту, над которым вы работаете.

Часто репозитории содержат файл README — файл с информацией о вашем проекте. Файлы README написаны на Markdown — удобном для чтения и записи языке форматирования обычного текста.

GitHub позволяет добавить файл README одновременно с созданием нового репозитория. GitHub также предлагает другие распространенные варианты, такие как файл лицензии, но сейчас можно не выбирать какой-либо из них.

Итак, ваш репозиторий hello-world может быть местом, где вы храните рабочие файлы, идеи, ресурсы и хотите ими поделиться и обсудить с другими.

1. Войдите на ресурс [github.com](https://github.com) своей учетной записью.
2. В правом верхнем углу страницы выберите знак + и в выпадающем списке нажмите «**Новый репозиторий**».




3. В поле «Имя репозитория» введите hello-world.
4. В поле «Описание» введите краткое описание. Например, введите «Этот репозиторий предназначен для практики GitHub Flow».
5. Выберите, будет ли ваш репозиторий **общедоступным** или **частным**. В данном случае укажите **Public** – общедоступный для всех пользователей.
6. Выберите **Добавить файл README**.

## 7. Нажмите **Создать репозиторий**.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (\*).



Owner \*  nikitaOsipof / Repository name \*

hello-world is available.

Great repository names are short and memorable. Need inspiration? How about [miniature-octo-telegram](#) ?

Description (optional)

Этот репозиторий предназначен для практики GitHub Flow

- ☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:

- ☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

### Шаг 2. Создайте ветку

Ветвление позволяет одновременно иметь разные версии репозитория.

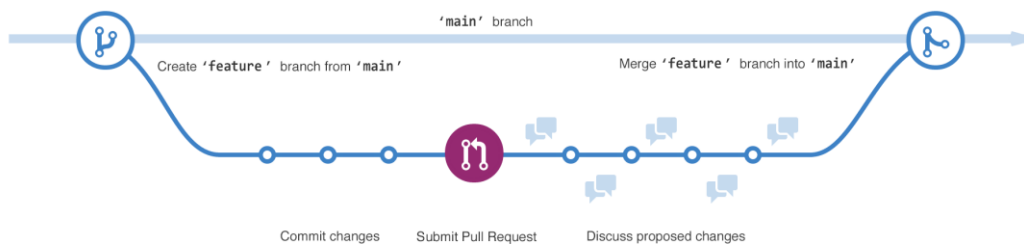
По умолчанию в вашем репозитории есть одна ветвь, main, которая считается окончательной. Вы можете создать дополнительные ветки main в своем репозитории.

Ветвление полезно, когда вы хотите добавить в проект новые функции без изменения основного источника кода. Работа, выполненная в разных ветвях, не будет отображаться в основной ветке, пока вы не объедините ее. Вы можете использовать ветки, чтобы экспериментировать и вносить изменения, прежде чем вносить их в main.

Когда вы создаете ветку на основе ветки main, вы создаете копию или снимок того main каким оно было на тот момент. Если кто-то другой внес изменения в ветку main, пока вы работали над ней, вы можете получить эти обновления.

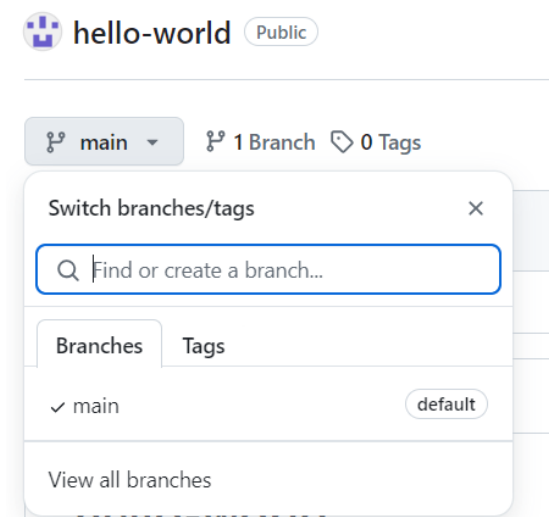
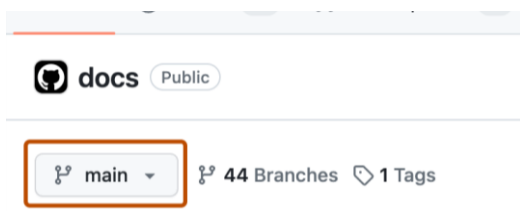
На этой диаграмме показано:

- Ветка main
- Новая ветка под названием feature
- Путь, который проходит ветка feature до того, как сливается с main

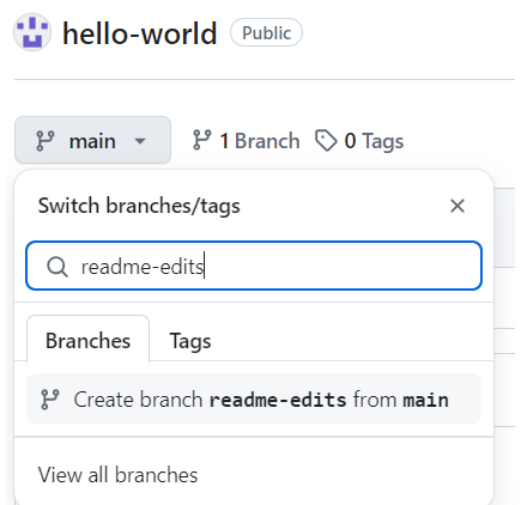


Для создания ветки выполните следующие действия:

1. Откройте вкладку «Код» вашего репозитория hello-world.
2. Над списком файлов щелкните раскрывающееся меню с надписью **main**.



3. Введите название ветки **readme-edits** в текстовое поле.
4. Нажмите «Create branch: **readme-edits** from **main**».



Теперь у вас есть две ветки: main и readme-edits. Прямо сейчас они выглядят совершенно одинаково. Далее вы добавите изменения в новую ветку readme-edits.

### Шаг 3. Внесите и зафиксируйте (commit) изменения.

Когда вы создали новую ветку на предыдущем шаге, GitHub перенес вас на кодовую страницу вашей новой ветки readme-edits, которая является копией main.

Вы можете вносить и сохранять изменения в файлы в вашем репозитории. На GitHub сохраненные изменения называются коммитами. С каждым коммитом связано сообщение о коммите, которое представляет собой описание, объясняющее, почему было сделано конкретное изменение. Сообщения о фиксации фиксируют историю ваших изменений, чтобы другие участники могли понять, что вы сделали и почему.

1. В созданной вами ветке readme-edits щелкните файл README.md.
2. Чтобы отредактировать файл, нажмите значок «карандаш».
3. В редакторе напишите о текущем проекте, произвольный текст.
4. Чтобы зафиксировать изменения нажмите **Commit changes**
5. В поле "Commit changes" напишите сообщение о фиксации, описывающее ваши изменения.
6. Нажмите **Commit changes**.

Эти изменения будут внесены только в файл README в вашей ветке readme-edits, поэтому теперь эта ветка содержит контент, отличный от main.

### Шаг 4. Откройте запрос на включение (pull request)

Теперь, когда у вас есть изменения в ветке main, вы можете открыть запрос на включение - pull request.

Запросы на включение — это основа совместной работы на GitHub. Когда вы открываете запрос на включение, вы предлагаете свои изменения и просите кого-нибудь просмотреть, извлечь ваш вклад и объединить его в свою ветку. Запросы на включение показывают различия в контенте обеих ветвей. Изменения, добавления и вычитания показаны разными цветами.



Как только вы сделаете коммит, вы сможете открыть запрос на включение и начать обсуждение, даже до того, как код будет закончен.

На этом этапе вы откроете запрос на включение в свой собственный репозиторий, а затем объедините его самостоятельно. Это отличный способ попрактиковаться в работе с GitHub перед работой над более крупными проектами.

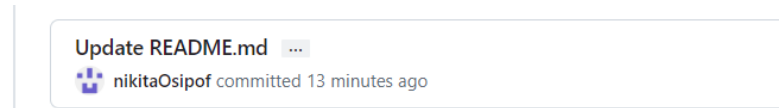
1. Перейдите на вкладку **«Pull request»** (Запросы на включение)- вашего репозитория hello-world.
2. Нажмите **New pull request (Новый запрос на включение)**.
3. В поле **«Example Comparisons»** выберите созданную вами ветвь readme-edits чтобы сравнить ее с main (оригиналом).

## Compare and review just about anything

Branches, tags, commit ranges, and time ranges. In the same repository and across forks.

Example comparisons	
 <code>readme-edits</code>	11 minutes ago
 <code>main@{1day}...main</code>	24 hours ago

4. Просмотрите изменения в различиях на странице сравнения и убедитесь, что это именно то, что вы хотите отправить.



Showing 1 changed file with 1 addition and 0 deletions.

1 README.md	
...	@@ -1,2 +1,3 @@
1	1 # hello-world
2	2 Этот репозиторий предназначен для практики GitHub Flow
3	+ Новая ветка создана для внесения изменений.

5. Нажмите **Create pull request (Создать запрос на включение)**.
6. Дайте вашему запросу название и напишите краткое описание ваших изменений. Вы можете включать смайлы и перетаскивать изображения и картинки.
7. Нажмите **Create pull request (Создать запрос на включение)**.

### Шаг 5. Объедините запрос на включение

На этом этапе вы объедините свою ветку `readme-edits` с веткой `main`. После объединения запроса на включение изменения в вашей ветке `readme-edits` будут включены в файл `main`.

Иногда запрос на включение может вносить изменения в код, которые конфликтуют с существующим кодом в `main`. Если возникнут какие-либо конфликты, GitHub предупредит вас о конфликтующем коде и предотвратит слияние до тех пор, пока конфликты не будут разрешены. Вы можете сделать коммит, разрешающий конфликты, или использовать комментарии в запросе на включение, чтобы обсудить конфликты с членами вашей команды.

В данном случае у вас не должно возникнуть конфликтов, поэтому вы готовы объединить свою ветку с основной.

1. В нижней части запроса на включение нажмите **«Merge pull request» (Объединить запрос на включение)**, чтобы объединить изменения в ветку `main`.
2. Для подтверждения объединения нажмите **Confirm merge**. Вы должны получить сообщение о том, что запрос успешно объединен и закрыт.



3. Теперь, когда ваш запрос на включение объединен и ваши изменения включены в `main`, вы можете безопасно удалить ветку `readme-edits` – для этого нажмите **Delete branch**. Если вы хотите внести дополнительные изменения в свой проект, вы всегда можете создать новую ветку и повторить этот процесс.
4. Вернитесь на вкладку **«Code»** вашего репозитория `hello-world`, чтобы просмотреть опубликованные изменения в ветке `main`.

## Итоги

Выполнив задания этого руководства, вы научились создавать проект и отправлять запросы на включение на GitHub. В рамках этого вы научились создавать репозиторий, создавать и управлять новой ветку, вносить изменения и фиксировать их на GitHub, открывать и объедините запрос на включение.

## Упражнение 2. Скачивание файлов из GitHub

В этом упражнении вы узнаете, как скачать файлы из GitHub, а также сможете понять разницу между загрузкой, клонированием (`cloning`) и ветвлением (`fork`).

GitHub.com является хранилищем для миллионов проектов программного обеспечения с открытым исходным кодом, которые можно копировать, настраивать и использовать для собственных целей.

Существуют различные способы получения копии файлов репозитория на GitHub.

Вы можете:

- **Скачать** моментальный снимок файлов репозитория в виде ZIP-файла на собственный (локальный) компьютер.
- **Клонировать** репозиторий на локальный компьютер с помощью `Git`.
- **Создать вилку (fork)** репозитория для создания нового репозитория на GitHub.

Каждый из этих методов имеет собственный вариант использования.

В этом руководстве рассматривается скачивание файлов репозитория на локальный компьютер. Например, если вы нашли интересное содержимое в репозитории на GitHub, скачивание — это простой способ получить копию содержимого без использования `Git` или применения управления версиями.

## Общие сведения о различиях между загрузкой, клонированием и вилкой

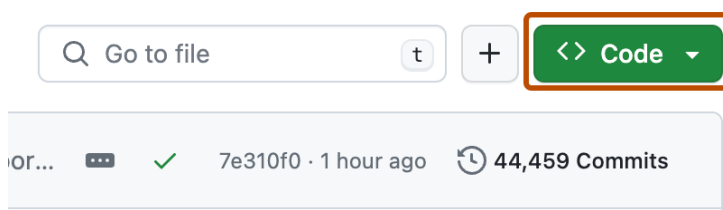
Термин	Определение	Вариант использования
Загрузка Download	Сохранение моментального снимка файлов репозитория на локальном компьютере.	Вы хотите использовать или настраивать содержимое файлов, но вы не заинтересованы в применении управления версиями.
Клонировать Clone	Сделать полную копию данных репозитория, включая все версии каждого файла и папки.	Вы хотите работать с полной копией репозитория на локальном компьютере, используя <code>Git</code> для отслеживания изменений и

		управления ими. Скорее всего, вы планируете синхронизировать эти локально внесенные изменения с помощью репозитория GitHub. Дополнительные сведения см. в разделе <a href="#">Клонирование репозитория</a> .
Вилка Fork	Создать новый репозиторий на GitHub, связанный с личной учетной записью, которая использует параметры кода и видимости с исходным ("вышестоящий") репозиторием.	Вы хотите использовать данные исходного репозитория в качестве основы для собственного проекта на GitHub. Кроме того, вы хотите использовать вилку для предложения изменений в исходном репозитории ("вышестоящий"). После создания вилки репозитория может потребоваться клонировать репозиторий, чтобы вы могли работать с изменениями на локальном компьютере. Дополнительные сведения см. в разделе <a href="#">Вилка репозитория</a> .

### Скачивание файлов репозитория

В этом руководстве мы будем использовать демонстрационный репозиторий ([octocat/Spoon-Knife](#)).

1. Перейдите к [октокату/ Spoon-Knife](#) .
2. Над списком файлов щелкните **Code**.



3. Щелкните **Download ZIP**.

### Итоги

Теперь у вас есть копия файлов репозитория, сохраненных в виде ZIP-файла на локальном компьютере. Вы можете изменять и настраивать файлы для собственных целей.

В следующем руководстве "[Отправка проекта в GitHub](#)" вы сможете узнать, как отправить собственные файлы в удаленный репозиторий на GitHub.

### Упражнение 3. Загрузка проекта на GitHub

В этом упражнении вы узнаете, как загрузить файлы вашего проекта на GitHub.

Загрузка файлов в репозиторий GitHub позволяет вам:

- **Применять контроль версий** при внесении изменений в файлы, чтобы история вашего проекта была защищена и управляема.
- **Сделать резервную копию** своей работы, потому что ваши файлы теперь хранятся в облаке.
- **Закрепить** репозиторий в своем личном профиле, чтобы другие могли видеть вашу работу.
- **Делиться** и обсуждать свою работу с другими, публично или конфиденциально.

#### Предварительные условия:

- У вас должна быть учетная запись GitHub.
- У вас должна быть группа файлов, которые вы хотите загрузить.

#### Шаг 1. Создайте новый репозиторий для вашего проекта.

Хорошая идея — создать новый репозиторий для каждого отдельного проекта, над которым вы работаете. Если вы пишете программный проект, группировка всех связанных файлов в новом репозитории упрощает обслуживание и управление базой кода с течением времени.

Действия по созданию нового репозитория описаны в упражнении *Базовые навыки*.

#### Шаг 2. Загрузите файлы в репозиторий вашего проекта.

После создания нового репозитория там есть только один файл — README.md, теперь вы загрузите несколько ваших файлов.

1. В правой части страницы выберите раскрывающееся меню **«Add file»**.
2. В раскрывающемся меню нажмите **«Upload files»**.
3. На своем компьютере откройте папку, содержащую вашу работу, затем перетащите все файлы и папки в браузер.
4. Внизу страницы в разделе *Commit changes* выберите "Commit directly to the main branch", затем нажмите **Commit changes**.

#### Шаг 3. Отредактируйте файл README для репозитория вашего проекта.

Файл README вашего репозитория обычно является первым элементом, который кто-то увидит при посещении вашего репозитория. Обычно он содержит информацию о том, о чем ваш проект и чем он полезен.

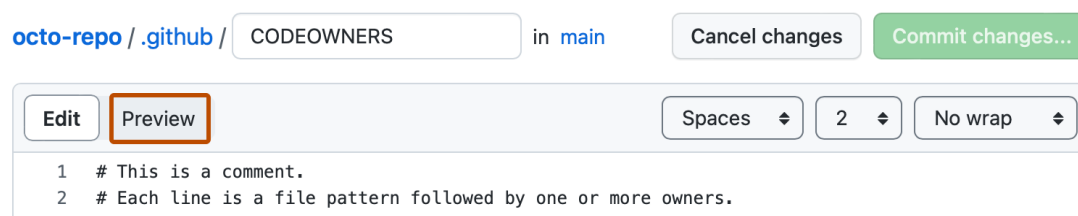
На этом этапе отредактируйте ваш проект README.md, чтобы он включал некоторую базовую информацию о вашем проекте.

1. В списке файлов нажмите, README.md чтобы просмотреть файл.

2. В правом верхнем углу просмотра файлов нажмите значок карандаша, чтобы открыть редактор файлов.
  - Вы увидите, что некоторая информация о вашем проекте была предварительно заполнена для вас. Например, вы должны увидеть имя репозитория и описание репозитория, которые вы заполнили на шаге 1, в строке 1 и строке 2.
3. Удалите существующий текст, кроме #, затем введите правильное название для вашего проекта.
  - Пример: # About my first project on GitHub.
4. Затем добавьте некоторую информацию о вашем проекте, например описание цели проекта или его основных функций.

Чтобы применить более сложное форматирование, например добавить изображения, ссылки и сноски, см. раздел « [Основной синтаксис написания и форматирования](#) ».

5. Над новым контентом нажмите «**Preview**».



6. Посмотрите, как будет отображаться файл после сохранения изменений, а затем вернитесь к «Edit».
7. Продолжайте редактировать и просматривать текст, пока не будете довольны содержанием README.
8. В правом верхнем углу нажмите «**Commit changes**».
9. В открывшемся диалоговом окне для вас предварительно заполнено сообщение о фиксации («Обновить README.md») и по умолчанию выбрана опция «Зафиксировать непосредственно в ветке main». Оставьте эти параметры без изменений и нажмите «**Commit changes**».

## Итоги

Вы создали новый репозиторий, загрузили в него несколько файлов и добавили файл README. Если вы установите для видимости репозитория значение «Общедоступный», репозиторий будет отображаться в вашем личном профиле, и вы сможете поделиться URL-адресом своего репозитория с другими.

Когда вы добавляете, редактируете или удаляете файлы непосредственно в браузере на GitHub, GitHub будет отслеживать эти изменения («фиксации»), поэтому вы можете начать управлять историей и развитием вашего проекта.

При внесении изменений помните, что вы можете создать новую ветку из ветки main вашего репозитория, чтобы можно было экспериментировать, не затрагивая

основную копию файлов. Затем, когда вы будете довольны набором изменений, откройте запрос на включение, чтобы объединить изменения в вашу ветку main.

### Следующие шаги

Большинство людей хотят продолжать работать над своими файлами локально (т. е. на своем компьютере), а затем постоянно синхронизировать эти локально внесенные изменения с этим «удаленным» (в облаке) репозиторием на GitHub. Существует множество инструментов, позволяющих это сделать, например **GitHub Desktop**. Чтобы начать, вам необходимо:

- **Установите GitHub Desktop.** Дополнительные сведения см. в разделе « [Начало работы с GitHub Desktop](#) ».
- **Клонируйте** удаленный репозиторий, чтобы иметь его копию на своем компьютере. Дополнительные сведения см. в разделе « [Клонирование и разветвление репозитория из GitHub Desktop](#) ».
- Постоянно **синхронизируйте** локальные изменения с этим удаленным репозиторием. Дополнительную информацию см. в разделе « [Синхронизация вашей ветки в GitHub Desktop](#) ».

## Практическое руководство 2. Работа с GitHub Desktop

В этом руководстве вы узнаете, как определить, проверить подлинность и настроить GitHub Desktop, чтобы участвовать в проектах непосредственно со своего компьютера.

**GitHub Desktop** — это бесплатное приложение с открытым кодом, которое помогает работать с кодом, размещенным в GitHub или других службах размещения Git.

С помощью GitHub Desktop можно выполнять команды Git, такие как фиксация и отправка изменений, в графическом пользовательском интерфейсе, а не с помощью командной строки. Дополнительные сведения см. в разделе [Сведения о GitHub Desktop](#).

Это руководство поможет вам приступить к работе с GitHub Desktop путем настройки приложения, проверки подлинности учетной записи, настройки основных параметров и введения основ управления проектами с помощью GitHub Desktop. После прохождения этого руководства вы сможете использовать GitHub Desktop для совместной работы над проектами и подключения к удаленным репозиториям.

- Прежде чем приступить к работе с GitHub, вам может быть полезно ознакомиться с Git и GitHub Desktop. Для получения дополнительных сведений см. следующие статьи.

- ["С помощью Git"](#)
- ["Знакомство с GitHub"](#)
- ["Начало работы с документацией по GitHub"](#)

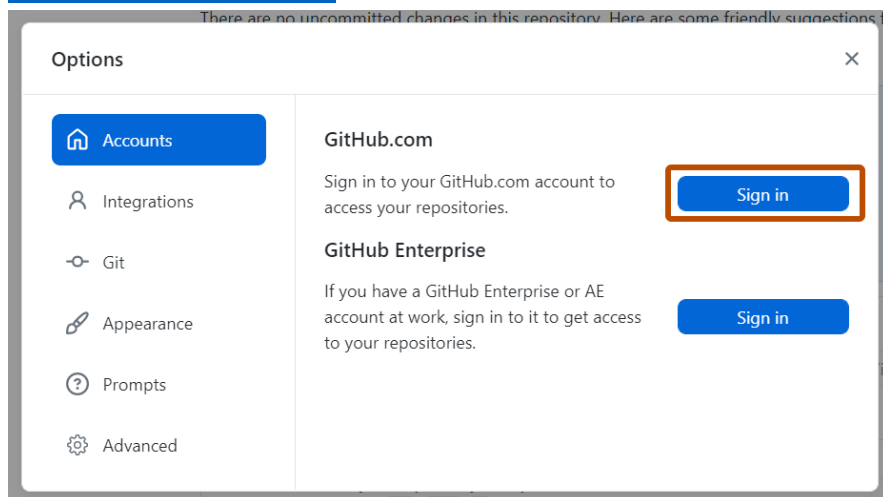
## 1. Установка и проверка подлинности

Установить GitHub Desktop можно в любой поддерживаемой операционной системе. Дополнительные сведения см. в разделе [Поддерживаемые операционные системы для GitHub Desktop](#).

Чтобы установить GitHub Desktop, перейдите на страницу скачивания [GitHub Desktop](#). Дополнительные сведения см. в разделе [Установка GitHub Desktop](#).

После установки GitHub Desktop вы можете выполнить проверку подлинности приложения с помощью учетной записи на GitHub или GitHub Enterprise. Проверка подлинности позволяет подключаться к удаленным репозиториям на GitHub или GitHub Enterprise.

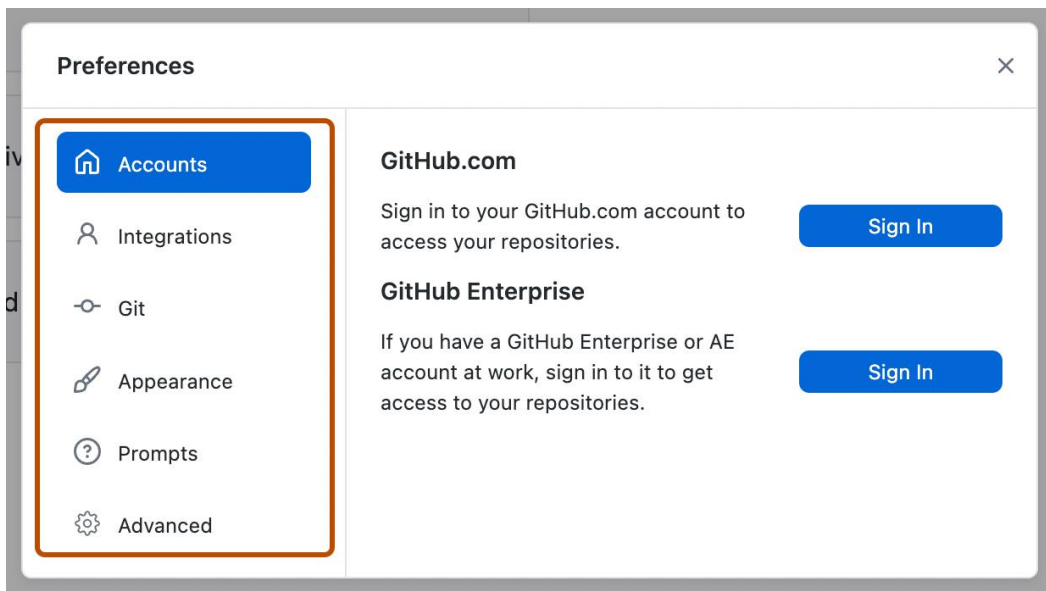
1. Прежде чем пройти проверку подлинности в GitHub или GitHub Enterprise, вам потребуется учетная запись. Дополнительные сведения см. в разделе "[Создание учетной записи на GitHub](#)".
2. В раскрывающемся меню "Файл" выберите **Параметры**. В окне параметров щелкните **Учетные записи** и выполните действия для входа. Дополнительные сведения о проверке подлинности см. в разделе "[Проверка подлинности на GitHub в GitHub Desktop](#)".



## 2. Настройка GitHub Desktop

После установки GitHub Desktop, вы можете настроить приложение в соответствии с вашими потребностями.

Вы можете подключать или удалять учетные записи для GitHub или GitHub Enterprise, выбрать текстовый редактор или оболочку по умолчанию, изменить конфигурацию Git, изменить внешний вид GitHub Desktop, настроить системные диалоговые окна и задать параметры конфиденциальности в окне "Настройки" GitHub Desktop. Дополнительные сведения см. в разделе [Настройка основных параметров в GitHub Desktop](#).



### 3. Участие в проектах с помощью GitHub Desktop

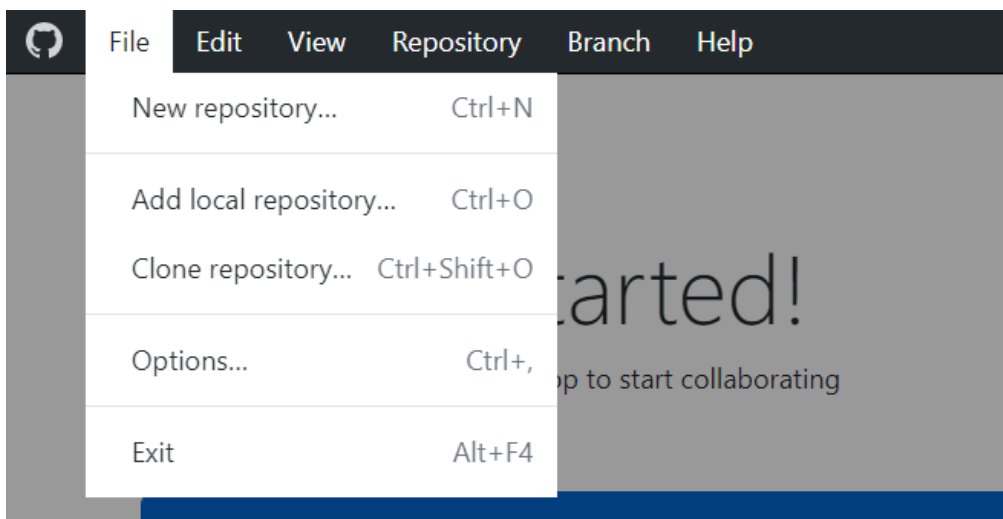
После установки, проверки подлинности и настройки приложения можно приступить к использованию GitHub Desktop. Вы можете создавать, добавлять или клонировать репозитории и использовать GitHub Desktop для управления своими вкладами в репозитории.

#### Создание, добавление и клонирование репозиторий

Вы можете создать репозиторий, выбрав **"Файл"** в строке меню "GitHub Desktop" и щелкнув **новый репозиторий....** Дополнительные сведения см. в разделе ["Создание первого репозитория с помощью GitHub Desktop"](#).

Вы можете добавить репозиторий с локального компьютера, выбрав **"Файл"** и нажав кнопку **"Добавить локальный репозиторий..."**. Дополнительные сведения см. в разделе ["Добавление репозитория с локального компьютера в GitHub Desktop"](#).

Вы можете клонировать репозиторий из GitHub, выбрав **файл** и щелкнув **"Клонировать репозиторий..."**. Дополнительные сведения см. в разделе ["Клонирование и создание ветки для репозиторий из GitHub Desktop"](#).



## Внесение изменений в ветвь

Вы можете использовать GitHub Desktop для создания ветви проекта. Ветви изолируют вашу работу по разработке от других ветвей в репозитории, чтобы вы могли безопасно экспериментировать с изменениями. Дополнительные сведения см. в разделе [Управление ветвями в GitHub Desktop](#).

После внесения изменений в ветвь их можно просмотреть в GitHub Desktop и выполнить фиксацию для отслеживания изменений. Дополнительные сведения см. в разделе [Фиксация и проверка изменений в проекте в GitHub Desktop](#).

Если вы хотите получить доступ к изменениям удаленно или поделиться ими с другими пользователями, вы можете отправить фиксации в GitHub. Дополнительные сведения см. в разделе [Отправка изменений в GitHub из GitHub Desktop](#).

## Совместная работа с помощью GitHub Desktop

Вы можете использовать GitHub Desktop, чтобы создавать проблемы и запросы на вытягивание для совместной работы над проектами с другими пользователями. Проблемы помогают отслеживать идеи и обсуждать возможные изменения проектов. Запросы на вытягивание позволяют делиться предлагаемыми изменениями с другими пользователями, получать отзывы и объединять изменения в проект. Дополнительные сведения см. в разделе "[Создание проблемы или запрос на вытягивание из GitHub Desktop](#)".

Вы можете просматривать собственные запросы на вытягивание и запросы других участников совместной работы в GitHub Desktop. Просмотр запроса на вытягивание в GitHub Desktop позволяет увидеть любые предлагаемые изменения и внести дополнительные изменения, открыв файлы и репозитории проекта в текстовом редакторе по умолчанию. Дополнительные сведения см. в разделе [Просмотр запроса на вытягивание в GitHub Desktop](#).

## Синхронизация локального репозитория

Когда вы вносите изменения в локальный репозиторий или когда другие пользователи вносят изменения в удаленные репозитории, вам потребуется синхронизировать локальную копию проекта с удаленным репозиторием. GitHub Desktop может поддерживать синхронизацию локальной копии проекта с удаленной версией путем отправки и извлечения фиксаций. Дополнительные сведения см. в разделе [Синхронизация ветви в GitHub Desktop](#).