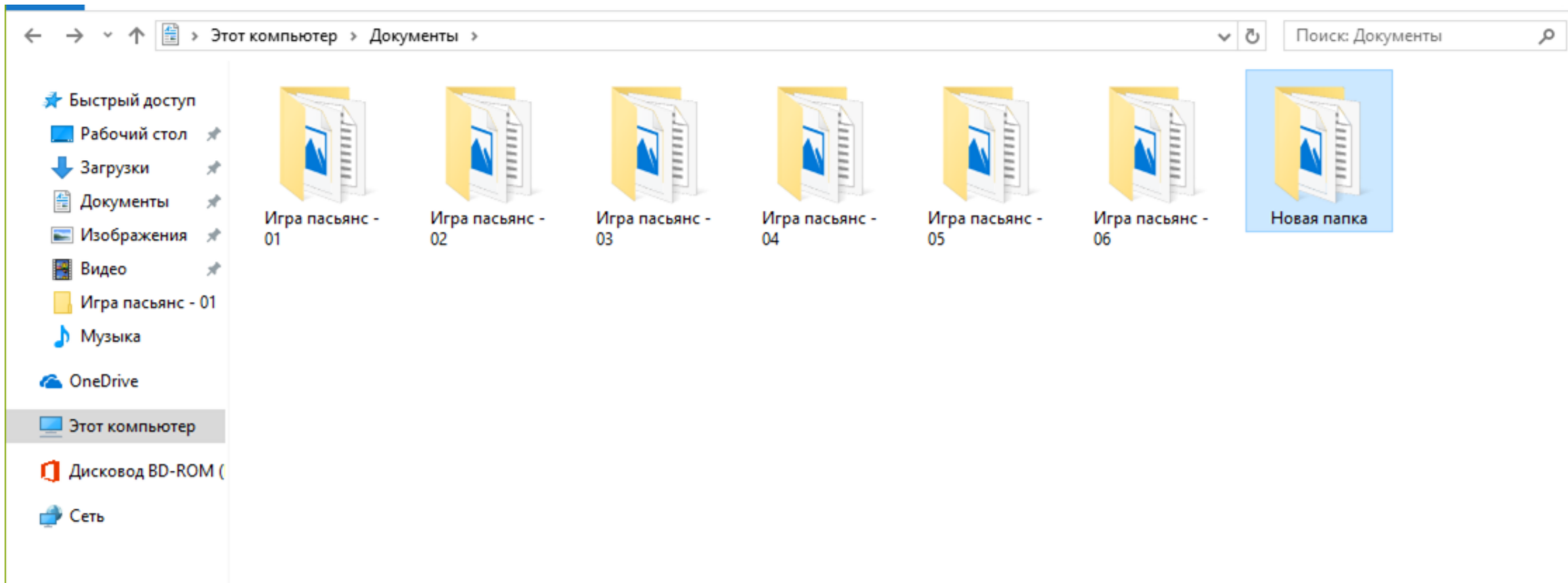


Управление версионностью программного
продукта (работа с системой контроля версиями
Git)

Резервные копии



Для чего нам нужен VCS

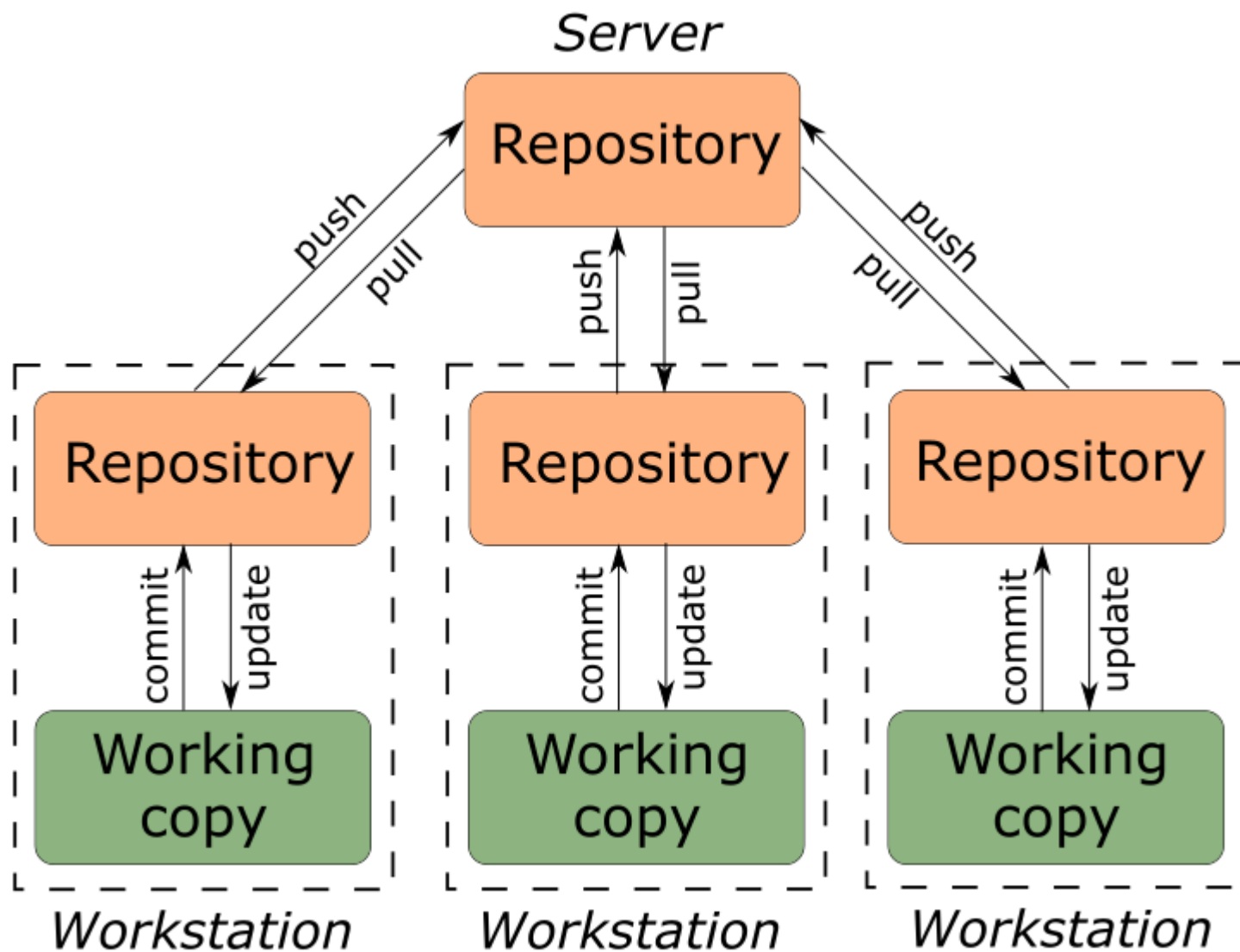
- Единое место хранения кода (для работы в команде)
- Синхронизация работы с командой (**объединение изменений** от разных разработчиков)
- Хранения истории разработки (истории изменений) с **описанием и авторством**
- Отмена неудачных изменений
- Альтернативные/экспериментальные реализации



Система контроля версий VCS

Система контроля версий (*от англ. Version Control System, VCS*) — это место хранения кода

Распределенные VCS (*Distributed Version Control System, DVCS*)



Распределенные VCS

- Гибкая работа с ветками
- Автономность (как каждого разработчика, так и от сервера вообще)
- Сборка артефактов отделена от разработки
- Разделены операции фиксации изменений (commit) и публикации изменений(push)



git



GitLab






GitHub



Bitbucket

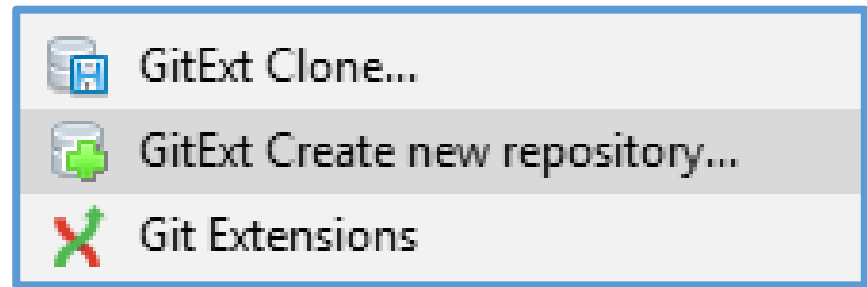
Рабочая директория

Имя	Дата изменения	Тип	Размер
 lib	21.02.2016 19:30	Папка с файлами	
 index.html	21.02.2016 19:30	Файл "HTML"	1 КБ
 README	21.02.2016 19:49	Файл	1 КБ

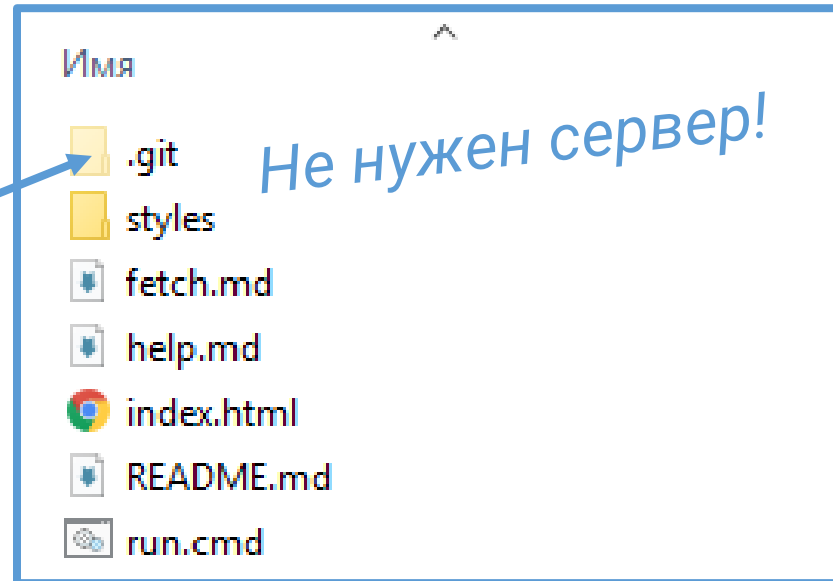
Репозиторий

Репозиторий – хранилище кода со всей историей изменений

`git init` – создать репозиторий для папки

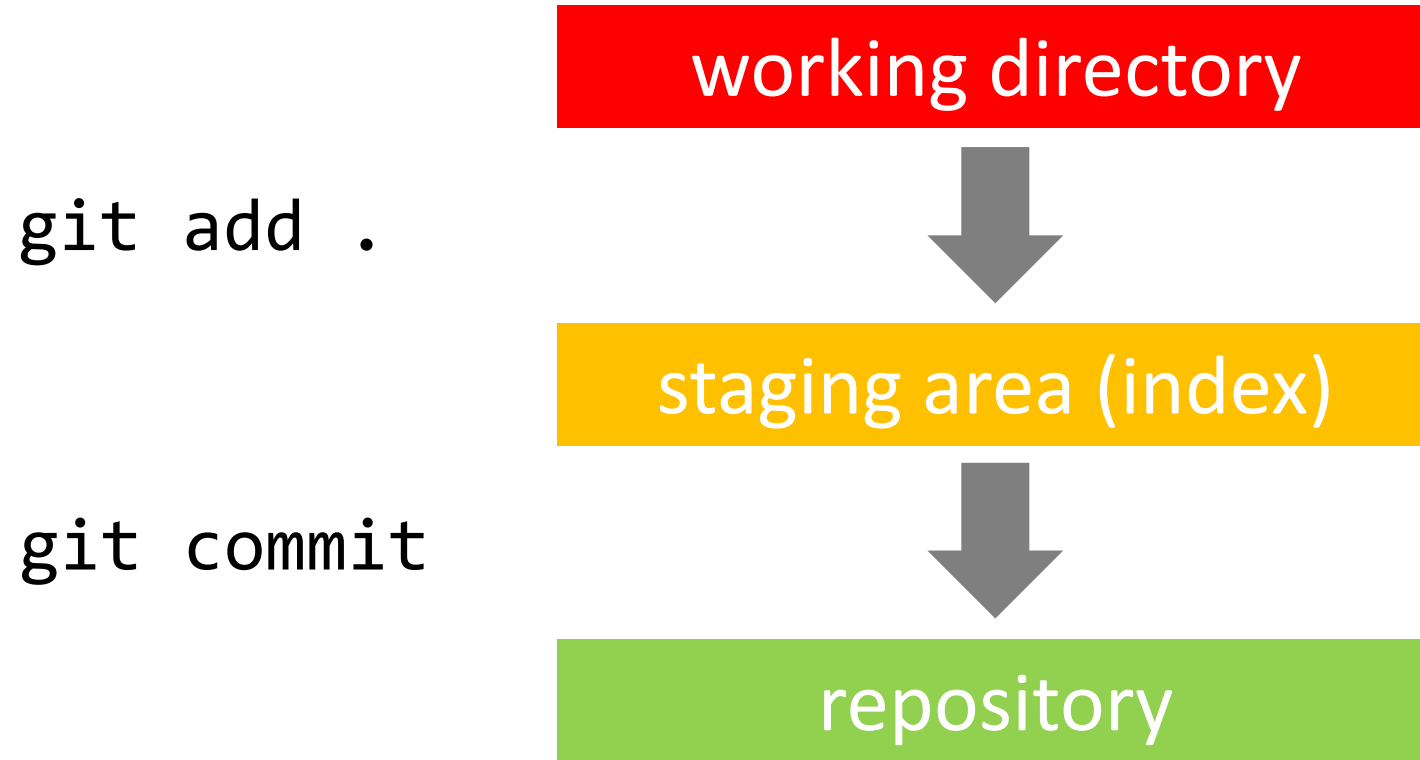


Репозиторий

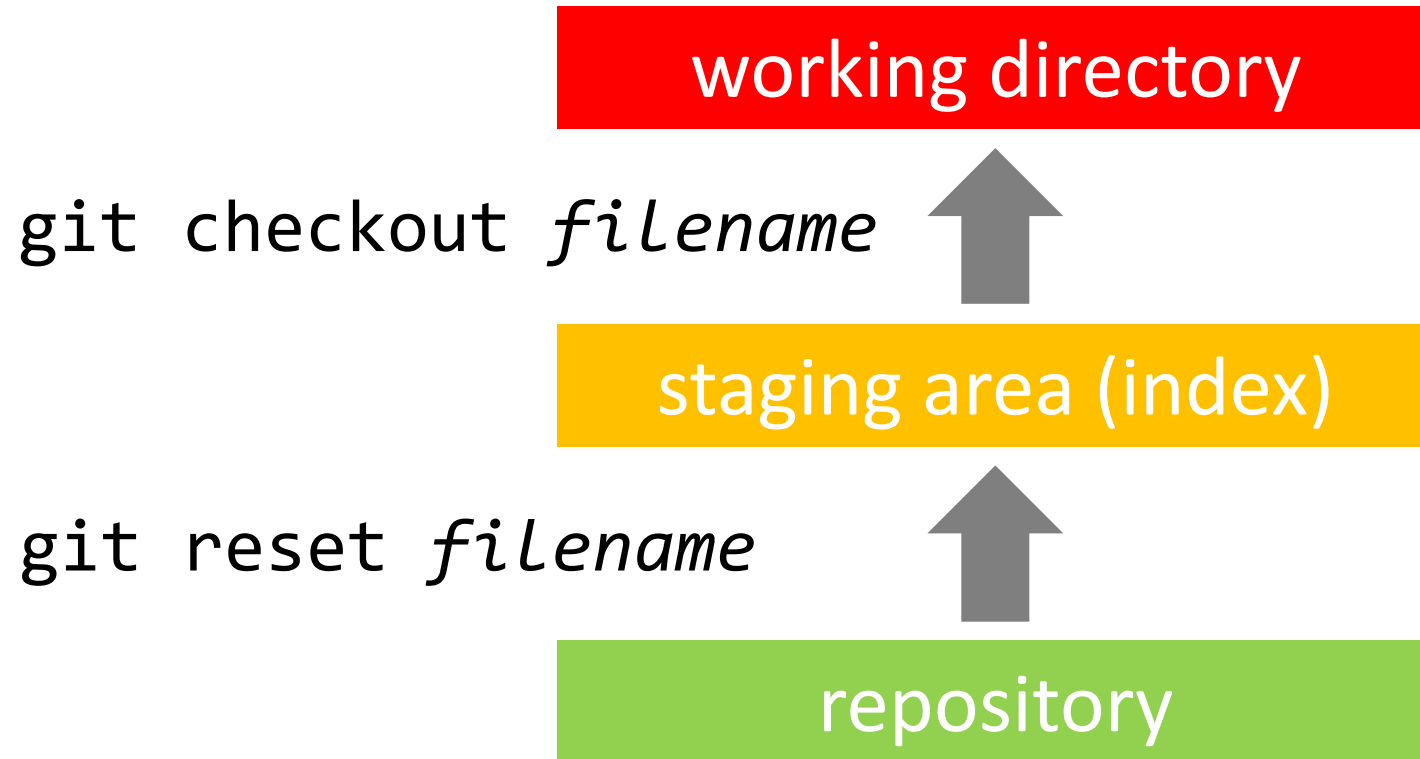


Рабочая
директория

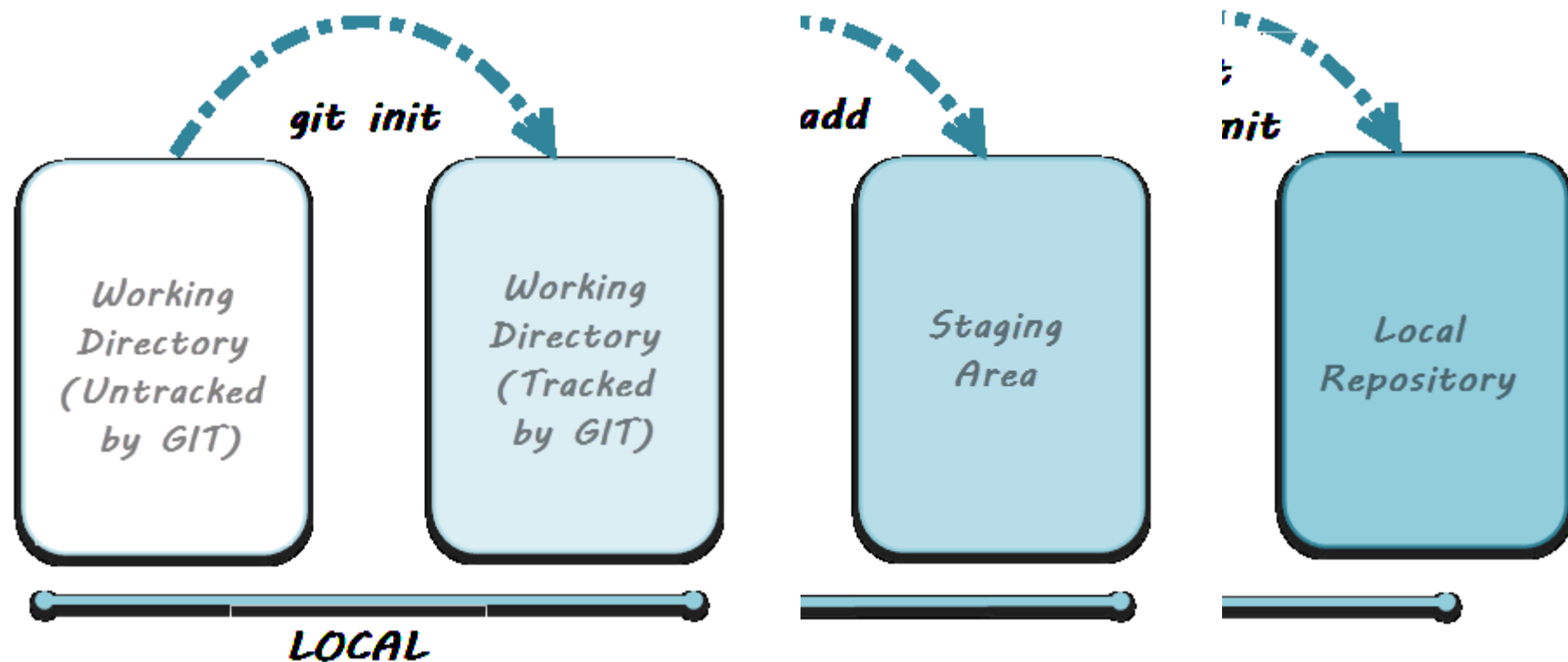
add to staging area and commit



checkout file or reset file



Жизненный цикл Git



HitHub

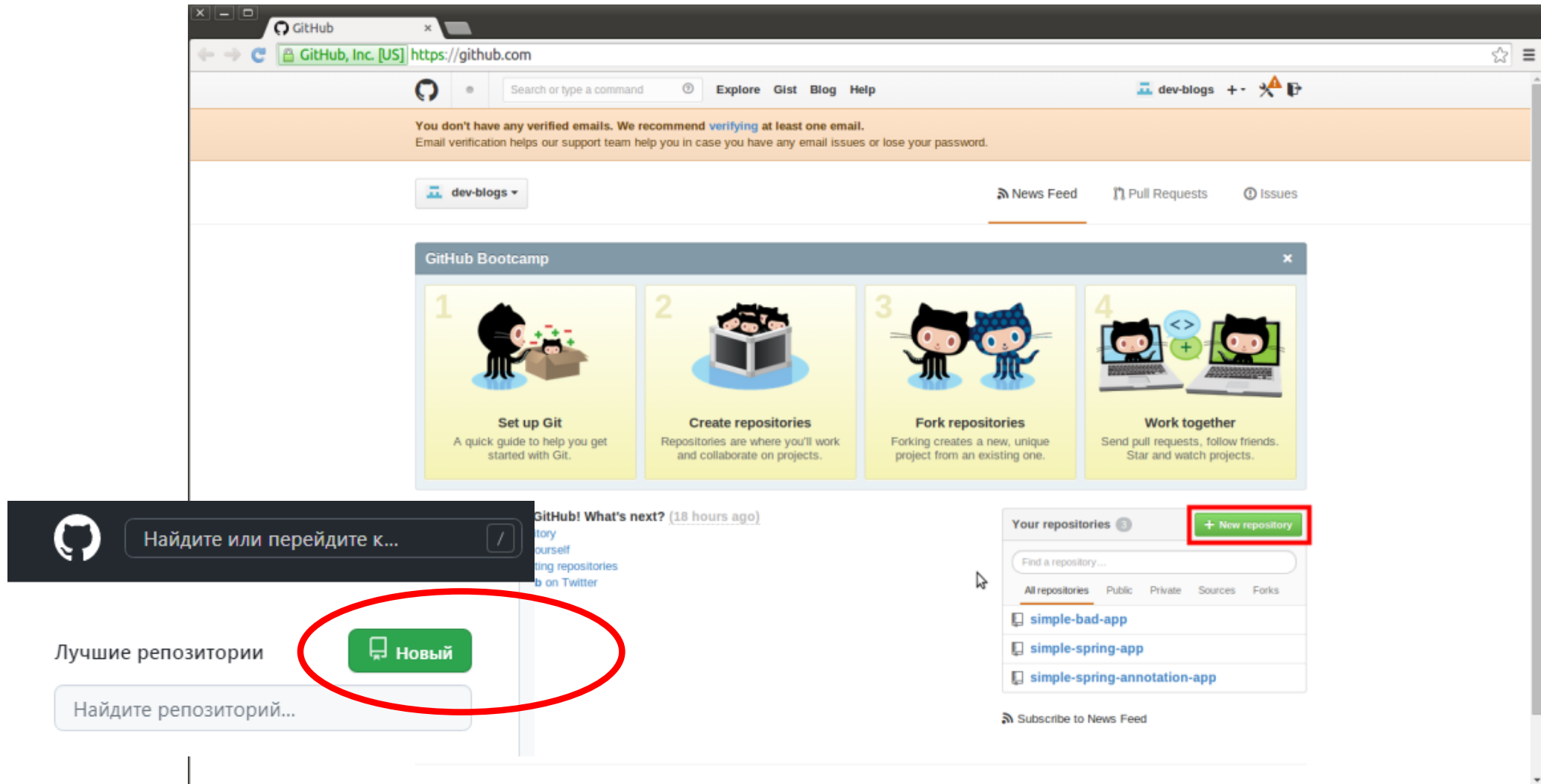
- <https://github.com/> - веб-сервис для хостинга IT-проектов и их совместной разработки, основанный на системе контроля версий Git



github
SOCIAL CODING

Создание репозитория

ШАГ 1



Создание репозитория

ШАГ 2

Создать новый репозиторий

Репозиторий содержит все файлы проекта, включая историю изменений. У вас уже есть репозиторий проекта в другом месте? [Импорт репозитория](#).

Владелец *



ГрибоваЭнн ▾


Имя репозитория *

/

Отличные имена репозитория короткие и запоминающиеся. Нужно вдохновение? Как насчет **книжной лампы** ?


Описание (необязательно)



 **Общественный**

Любой пользователь Интернета может видеть этот репозиторий. Вы выбираете, кто может совершить.



 **Частный**

Вы выбираете, кто может видеть и фиксировать этот репозиторий.

Инициализируйте этот репозиторий с помощью:

Пропустите этот шаг, если вы импортируете существующий репозиторий.



Добавьте файл README

Здесь вы можете написать подробное описание вашего проекта. [Узнать больше](#).

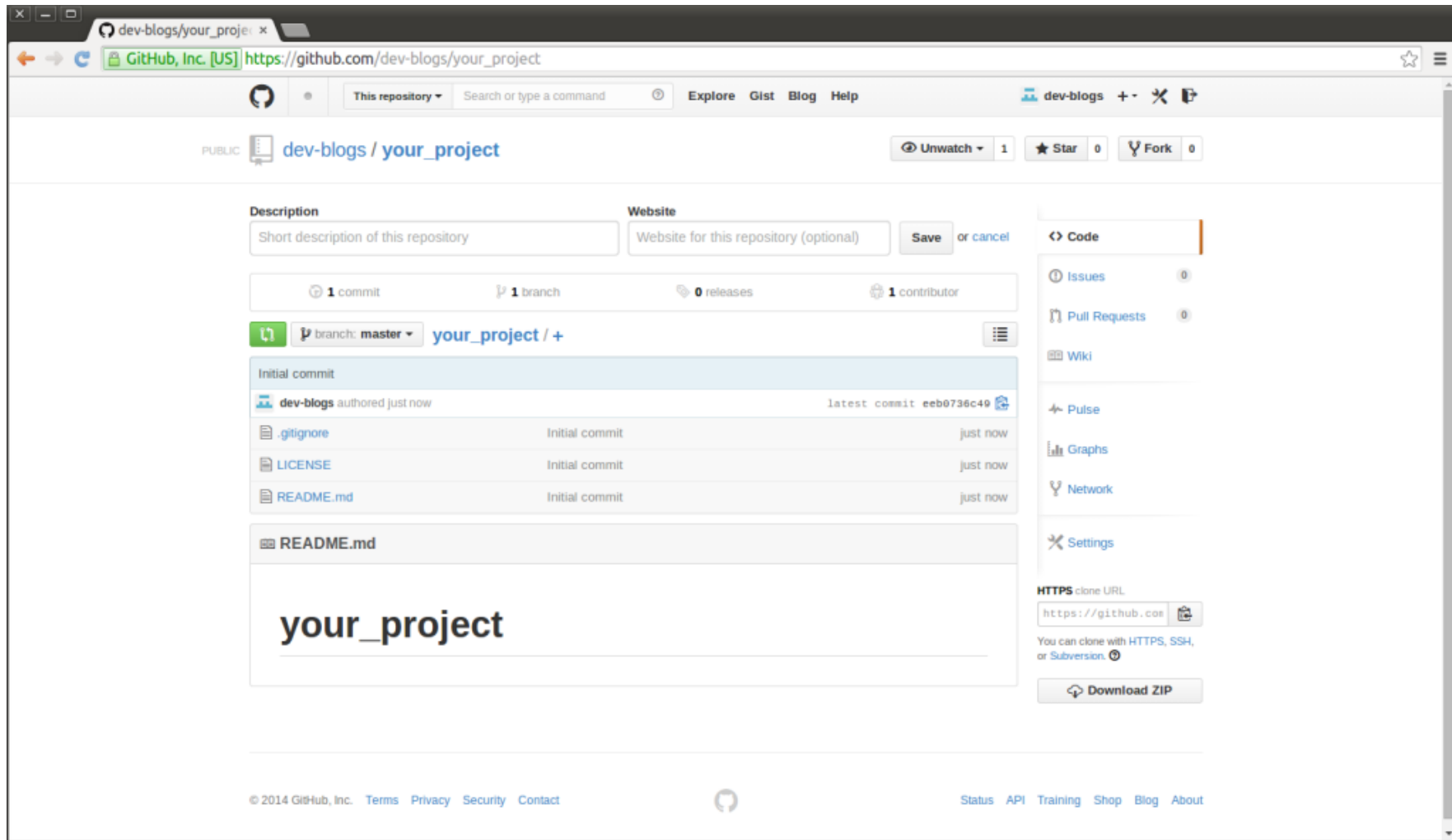
Добавить .gitignore

Выберите, какие файлы не отслеживать из списка шаблонов. [Узнать больше](#).

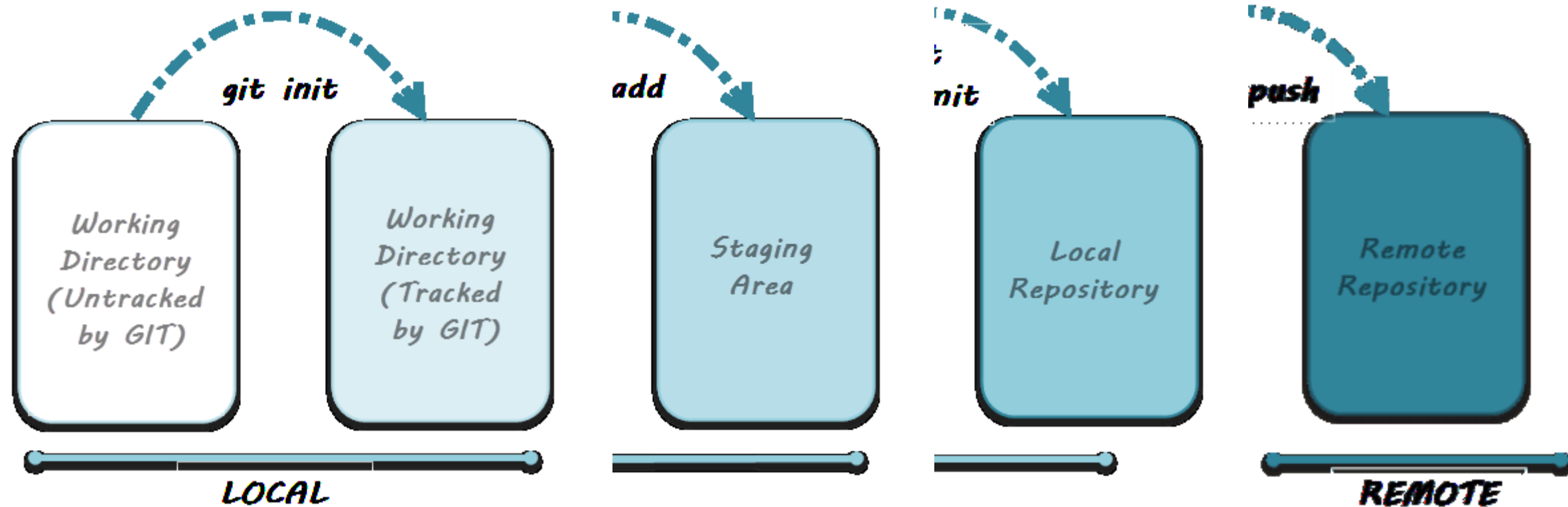
.gitignore шаблон: Никто ▾

Создание репозитория

ШАГ 3



Жизненный цикл Git



Команды для отправки изменений в удаленный репозиторий

git init

git add .

git commit -m "текст сообщения"

git remote add origin "ссылка_на_репозиторий"

git branch -M "имя_ветки"

git push -u origin "имя_ветки"

Команды для отправки изменений в удаленный репозиторий

1. Подтвердите существование вашего ЛОКАЛЬНОГО репозитория Git.

git init

git add .

git commit -m "текст сообщения"

git log

2. Создайте новый пустой репозиторий Git на удаленном сервере.
3. Получите URL-адрес удаленного добавления git для удаленного репозитория и при необходимости добавьте учетные данные.
4. Запустите команду **git remote add origin** из локального репозитория с параметром **--set-upstream** и именем активной ветки для отправки.

git remote add origin "ссылка_на_репозиторий"

git push -u origin "имя_ветки"

2. Просмотрите отправленные файлы в удаленном репозитории Git, чтобы убедиться, что команды **git remote add** и **push** выполнены успешно.

Подытожим!!

Работа с локальным репозиторием

- Команда `add` добавляет измененные файлы в stage
- Команда `rm` помечает файл в stage как удаленный
- Команда `reset` сбрасывает изменения в текущем stage
- Команда `commit` сохраняет текущий stage в локальный репозиторий

Работа с удаленным репозиторием

- Команда `clone` клонирует репозиторий и создаёт рабочую копию
- Команда `push` отправляет изменения в удаленный репозиторий
- Команда `pull` забирает изменения указанной ветки из удаленного репозитория и сливает их в текущую ветку
- Команда `fetch` забирает все изменения из удаленного репозитория

Работа с ветками

- Команда `branch` создаёт ветку
- Команда `checkout` переключает рабочую копию на другую ветку
- Команда `merge` сливает изменения веток
- Команда `stash` помещает изменения из stage во временное хранилище и сбрасывает рабочую копию

Полезные ссылки

- <https://githowto.com/ru> -Обучающий курс по работе с Git
- <http://proselyte.net/tutorials/git/introduction/> - русскоязычная документация, начальные шаги
- <https://git-scm.com/book/en/v2> - основательная документация, почти на все случаи жизни

Работа с HitHub из IDE



github
SOCIAL CODING

Настройка доступа к репозиторию

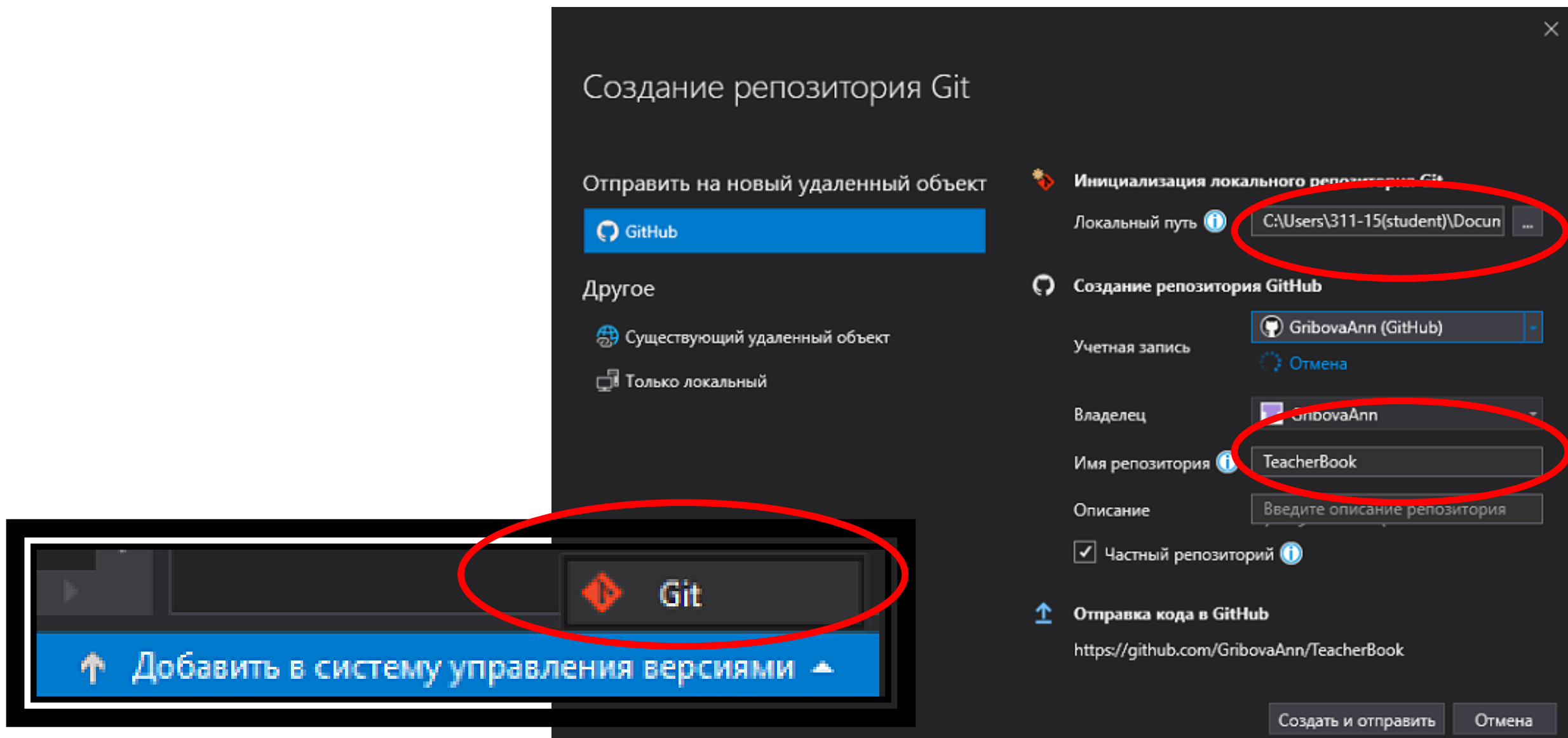
The screenshot shows the GitHub repository settings page. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. The 'Settings' link is circled in red. Below the navigation bar is the 'Danger Zone' section, which contains four actions:

- Change repository visibility**: This repository is currently private. [Change visibility](#)
- Transfer ownership**: Transfer this repository to another user or to an organization where you have the ability to create repositories. [Transfer](#)
- Archive this repository**: Mark this repository as archived and read-only. [Archive this repository](#)
- Delete this repository**: Once you delete a repository, there is no going back. Please be certain. [Delete this repository](#)

Two green callout bubbles provide additional context:

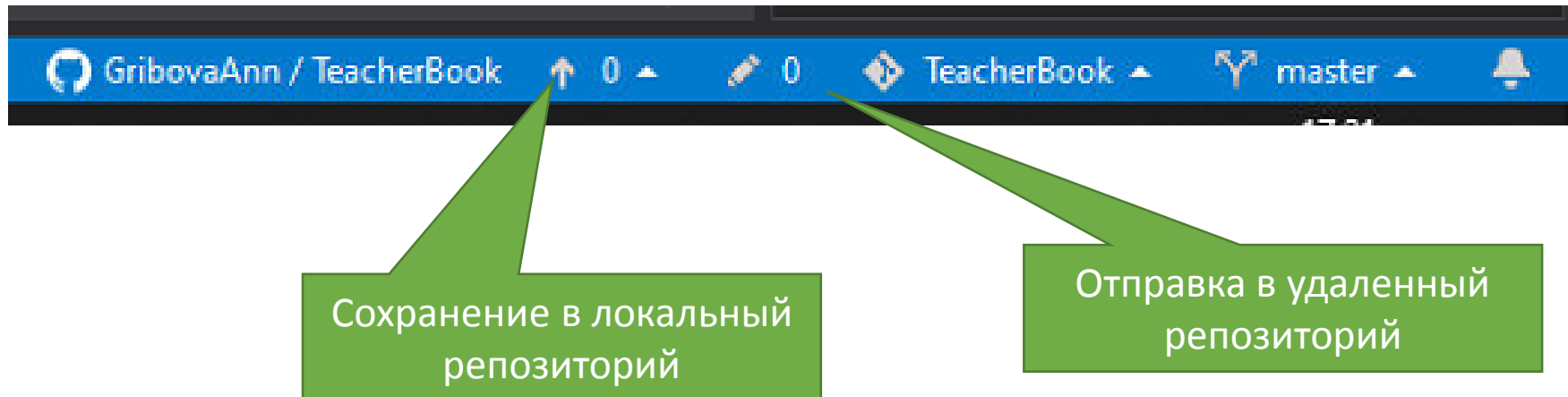
- A bubble pointing to the 'Settings' tab: **Изменение доступа к репозиторию**
- A bubble pointing to the 'Delete this repository' button: **Удаление репозитория**

Создание связи между локальным и удаленным репозиторием

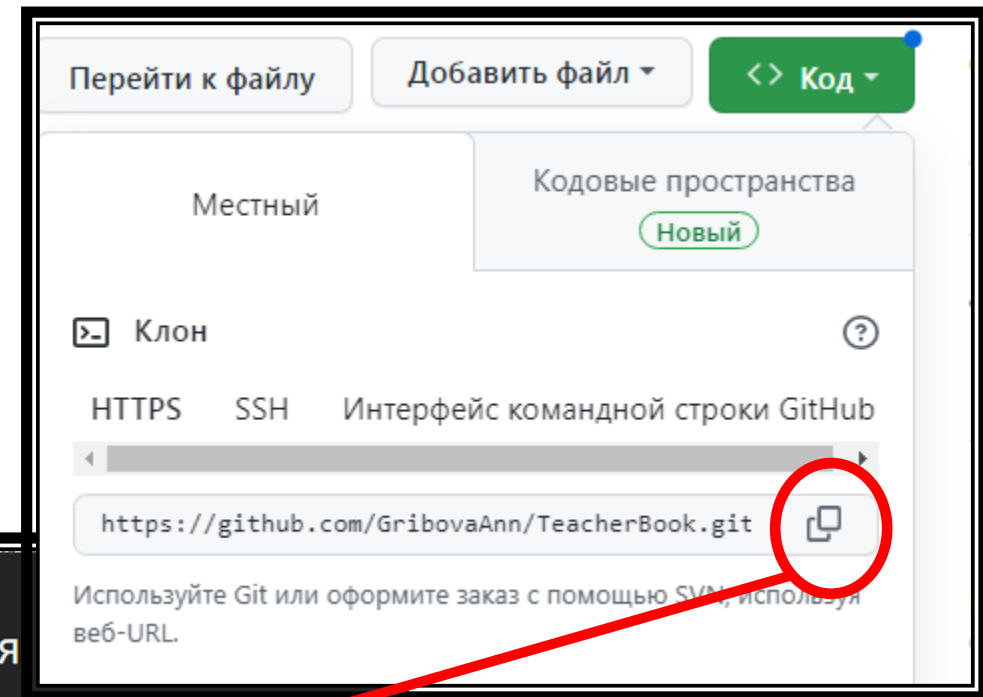
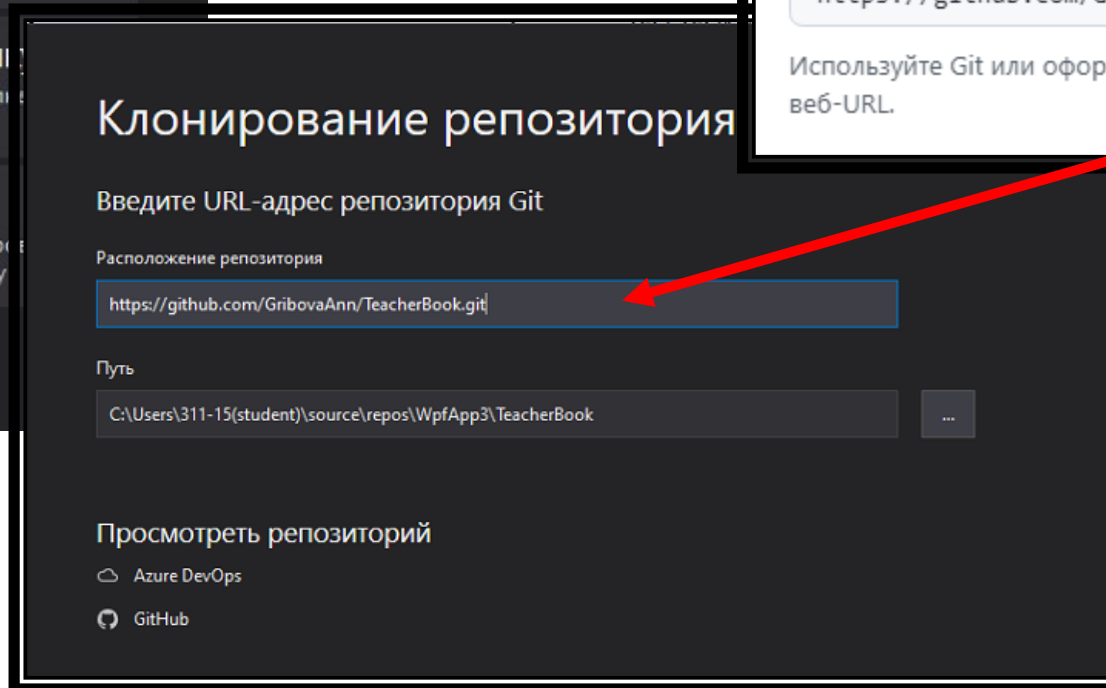
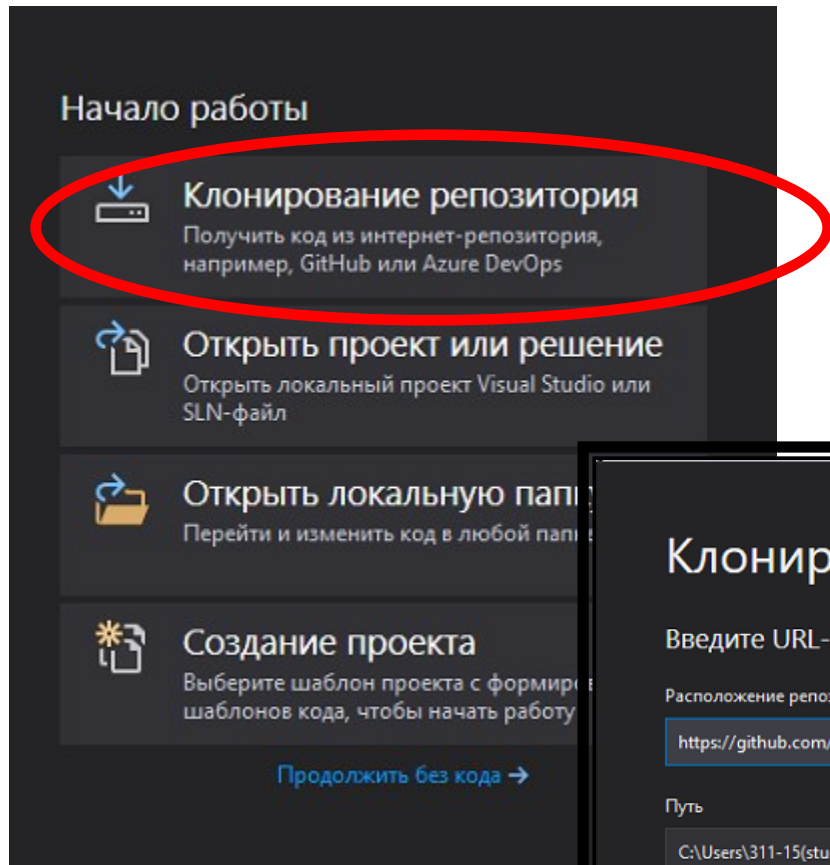


Фиксация изменений

Фиксация изменений прошла успешно, если изменения успешно зафиксированы и в локальном, и в удаленном репозитории

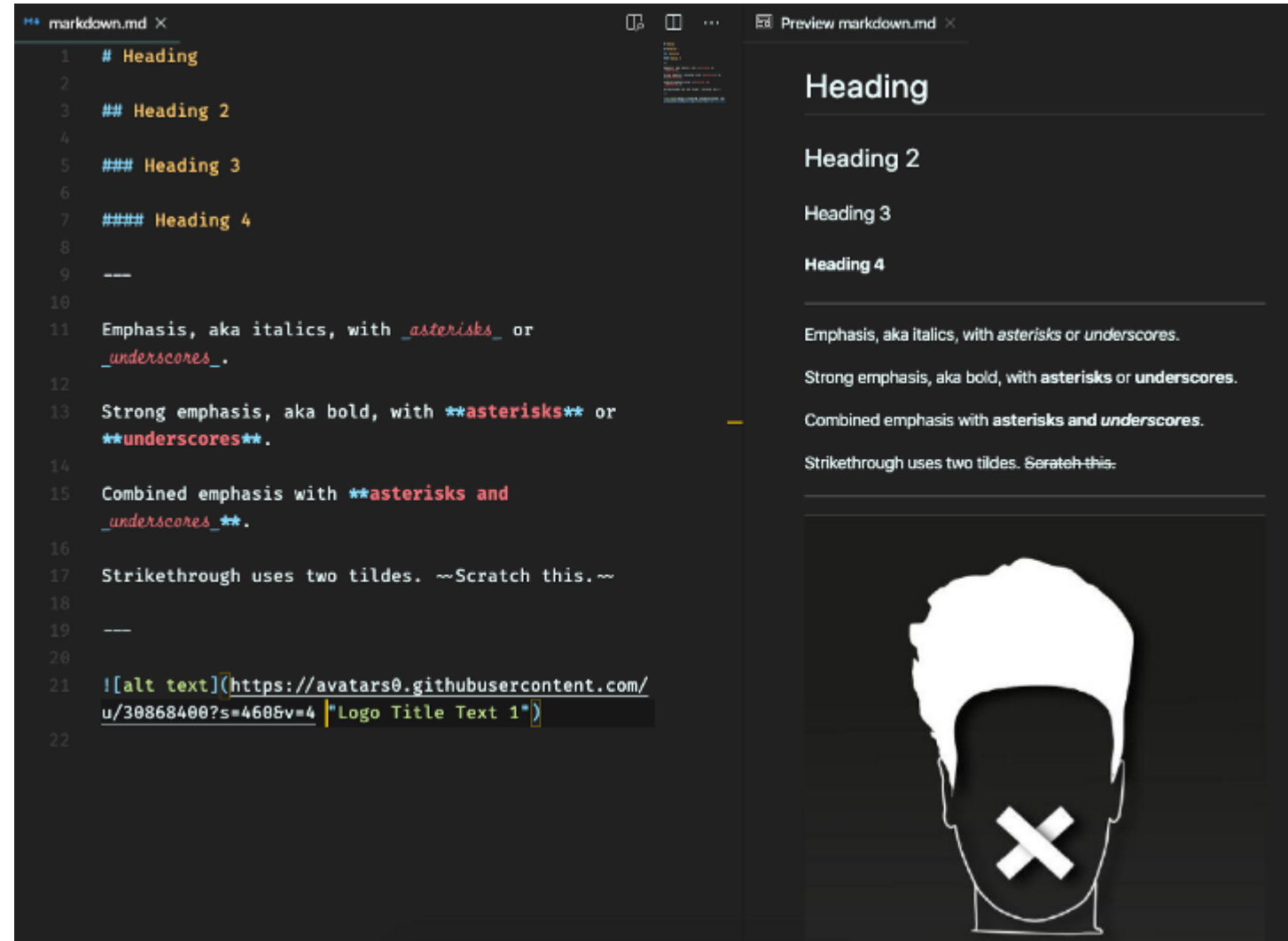


Клонирование репозитория с Github



Файл Readme.md

1. Название проекта
2. Описание проекта, основная цель проекта
3. Как установить ваш проект.
Описать шаги, необходимые для установки вашего проекта.
Пошаговое описание того, как запустить среду разработки.
4. Как использовать ваш проект (можете вставить скриншоты, чтобы показать примеры работающего проекта)
5. Команда разработчиков (перечислите своих соавторов / членов команды, включите ссылки на их профили).



Образец файла Readme.md

Заголовок проекта

Один абзац описания проекта находится здесь

Начало работы

Эти инструкции предоставят вам копию проекта и помогут запустить на вашем локальном компьютере для разработки и тестирования.

Необходимые условия

Что нужно для установки программного обеспечения и как его установить

```Предоставьте примеры```

### Установка

Пошаговая серия примеров, которые говорят, что вы должны запустить

Скажи, какой будет шаг

```Приведи пример```

И повтори

```Пока не закончил```

Завершите пример получением некоторых данных о системе или использования их для небольшой демонстрации

## Авторы

\* \*\*Billie Thompson\*\* - \*Initial work\* - [PurpleBooth](<https://github.com/PurpleBooth>)

See also the list of [contributors](<https://github.com/your/project/contributors>) who participated in this project.



Gogs

# Авторизуйтесь в Gogs под своей учетной записью.

[Главная](#)[Обзор](#)[Помощь](#)[Вход](#)

## Вход

Имя пользователь или  
E-mail \*

Пароль \*

☐ Запомнить меня

Вход

[Забыли пароль?](#)



На верхней панели нажмите «+» и выберите «Новый репозиторий».

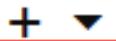


Панель управления

Задачи

Запросы на слияние

Обзор



TheWorst ▼

+ Новый репозиторий


Новая миграция



TheWorst запустил(а) master в TheWorst/STM

Укажите имя репозитория, задайте видимость репозитория, нажмите на «Создать репозиторий».

### Новый репозиторий

Владелец \*  s10

Имя репозитория \*

Лучшие названия репозитория коротки, запоминаемы и уникальны.


Видимость ☒ Личный репозиторий

Описание


---

.gitignore

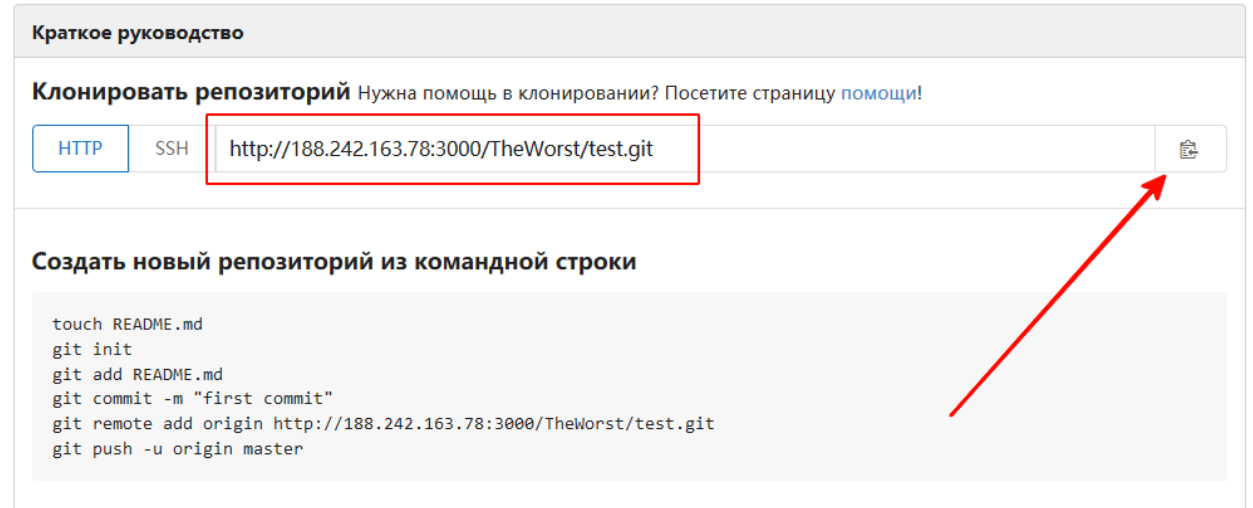
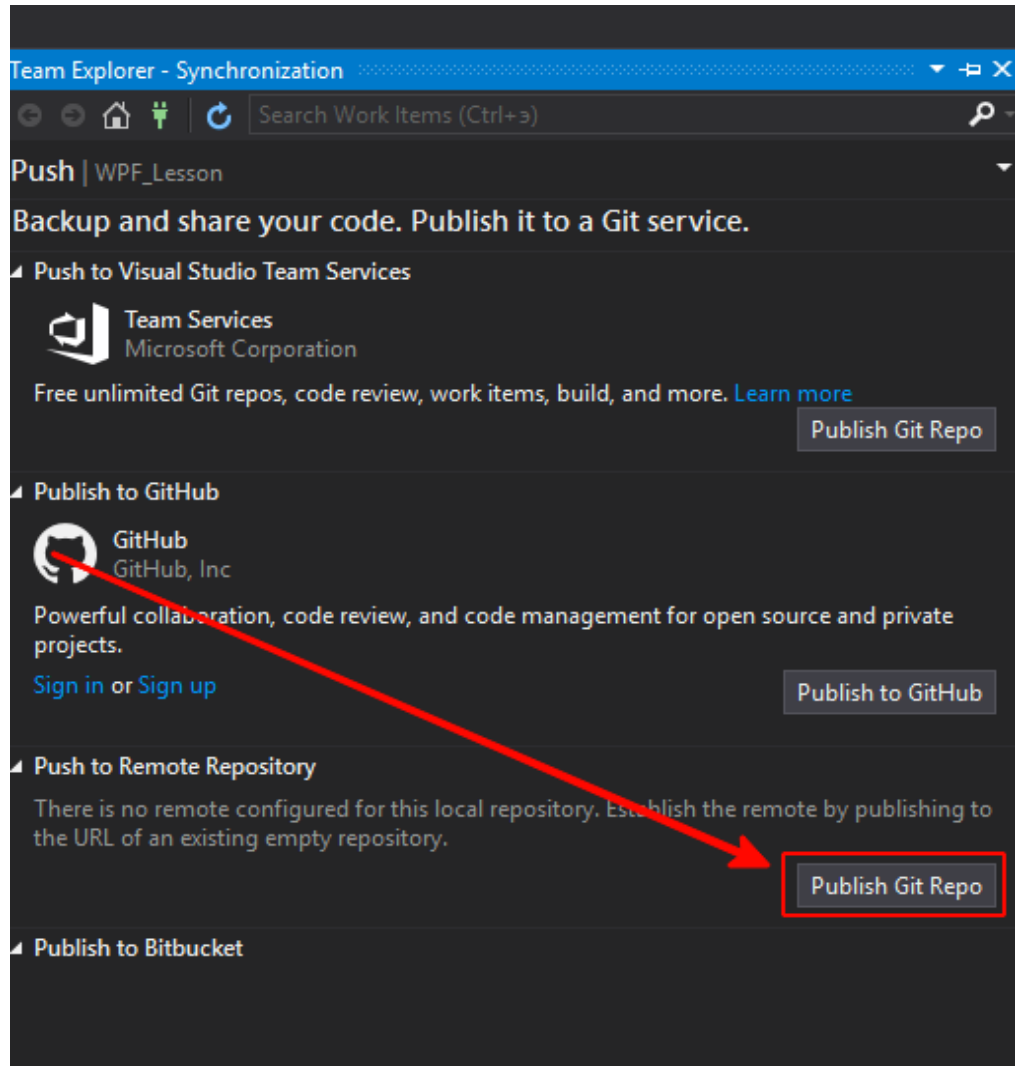
Лицензия

Readme 

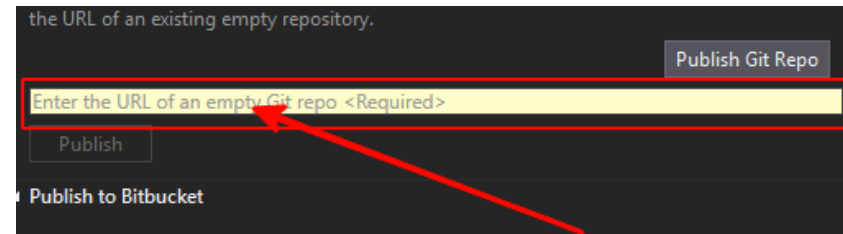
☐ Инициализировать этот репозиторий выбранными файлами и шаблоном



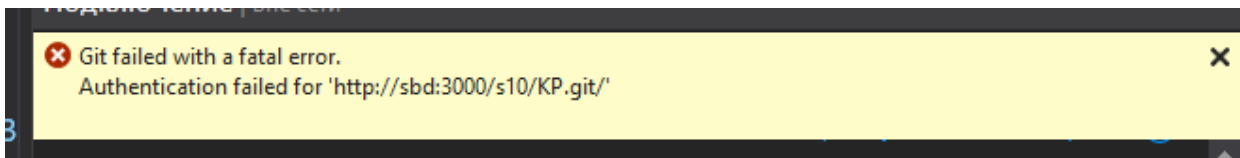
# Создание связи между локальным и удаленным репозиторием



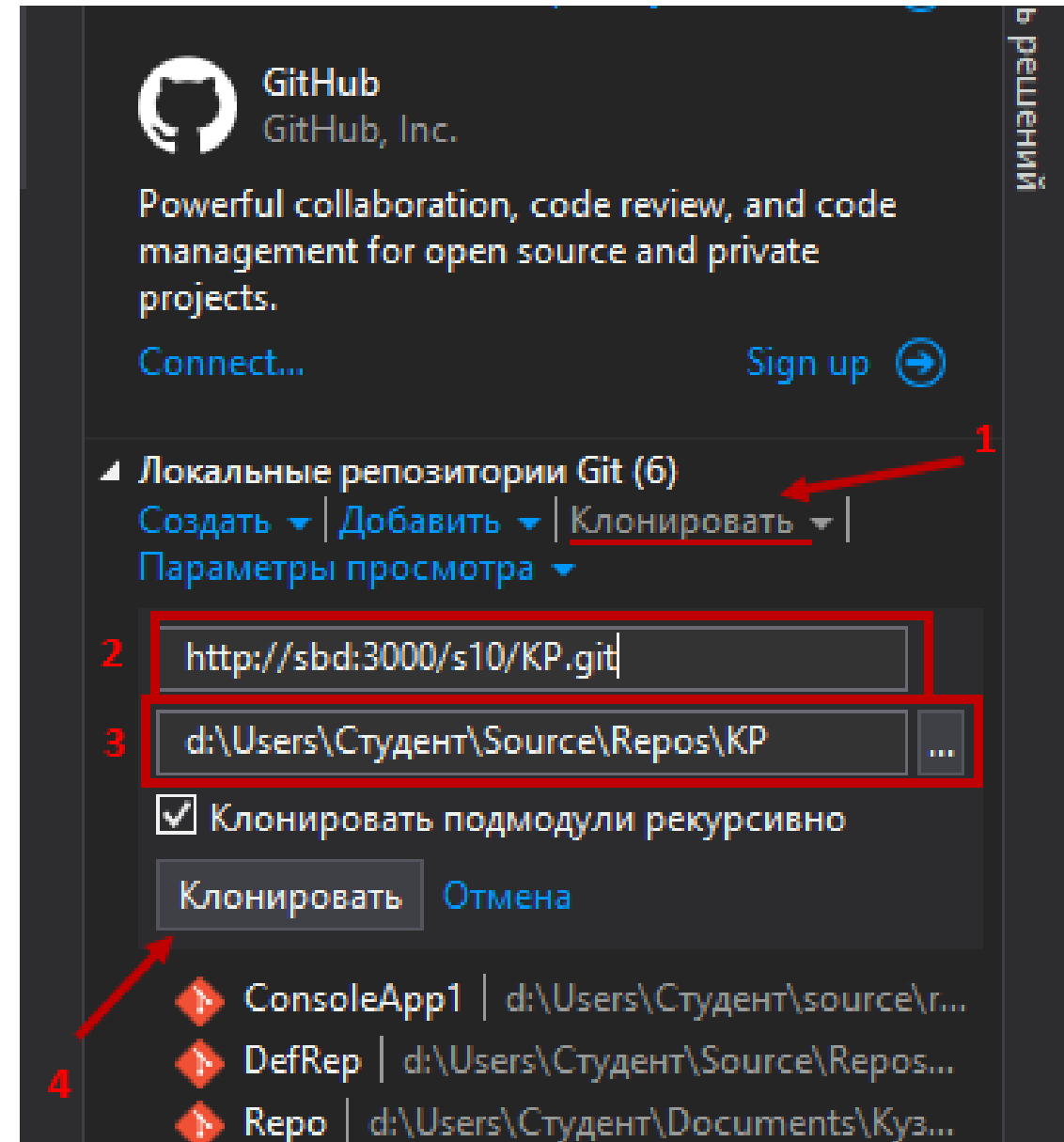
1. Скопируйте в Gogs ссылку на репозиторий
2. Вставьте ссылку на репозиторий в поле подключения удаленного репозитория и нажмите кнопку «Publish».



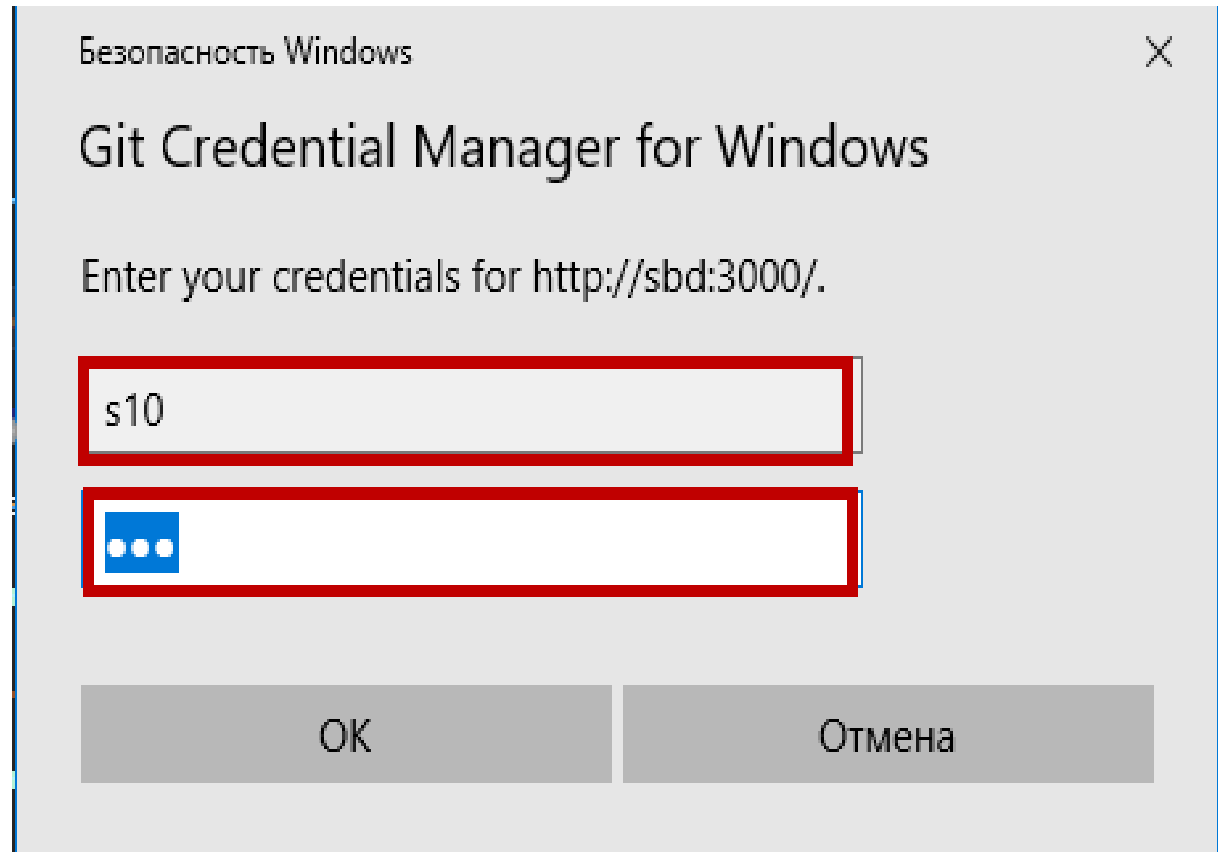
1. Перейдите на вкладку «Клонировать»
2. заполните поле « ссылка на репозиторий»
3. заполните поле « папка для копирования с удаленного репозитория (клонирования)»
4. нажмите на кнопку «клонировать»



Если возникает такая ошибка, нажмите на кнопку «клонировать» повторно.



Заполните поля Логина и Пароля в возникшем окне авторизации данными, которые были указаны при регистрации на GOGS



Безопасность Windows

Git Credential Manager for Windows

Enter your credentials for <http://sbd:3000/>.

s10

...

OK Отмена

The image shows a Windows security dialog box titled "Безопасность Windows" (Windows Security). The main title is "Git Credential Manager for Windows". Below it, it says "Enter your credentials for http://sbd:3000/". There are two input fields: the first contains the text "s10", and the second is a password field with three dots and a blue icon. Both fields are highlighted with a red border. At the bottom, there are two buttons: "OK" and "Отмена" (Cancel).

Зайдите во вкладку меню «Параметры».

Выберите вкладку «Параметры репозитория» и заполните поля «Имя пользователя» и «Адрес электронной почты», которые были указаны при регистрации на GOGS

The image shows a composite of three screenshots from the Team Explorer application, illustrating the steps to configure repository parameters.

- Bottom-left screenshot:** Shows the main Team Explorer interface with the 'Параметры' (Parameters) menu item circled in red.
- Top-left screenshot:** Shows the 'Параметры' (Parameters) sub-menu with 'Параметры репозитория' (Repository Parameters) circled in red.
- Right screenshot:** Shows the 'Параметры Git-репозитория' (Git Repository Parameters) dialog box. The 'Имя пользователя' (Username) field is filled with 's10' and the 'Адрес электронной почты' (Email) field is filled with 'user16@user16.ru'. The 'Обновить' (Update) button is highlighted with a red arrow.

# Фиксация изменений

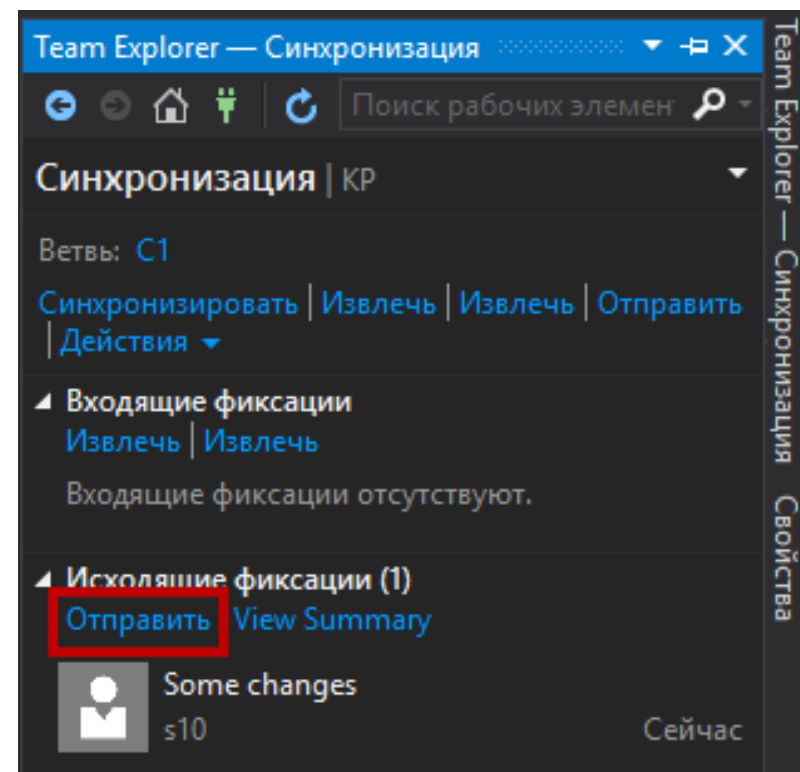
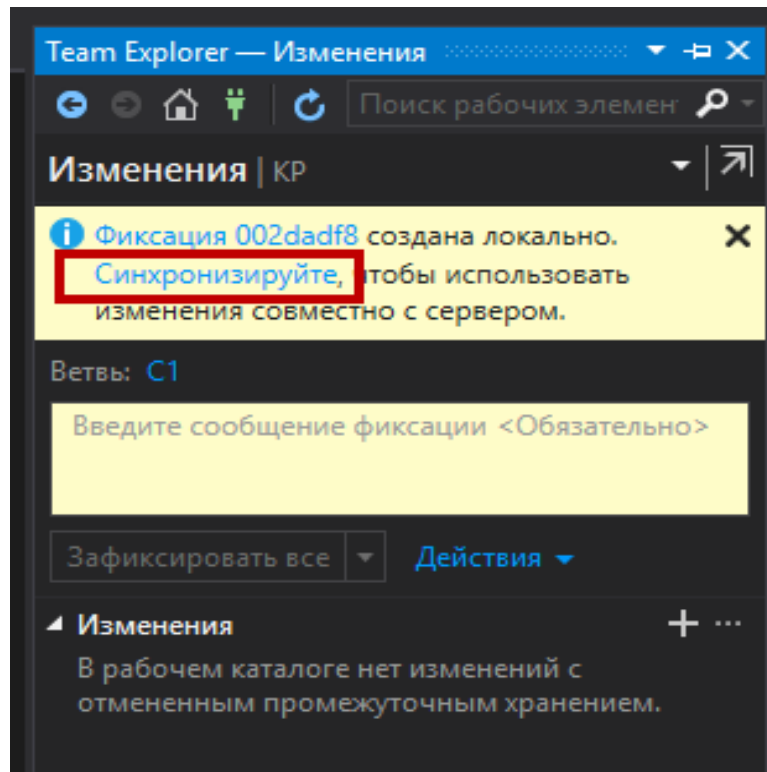
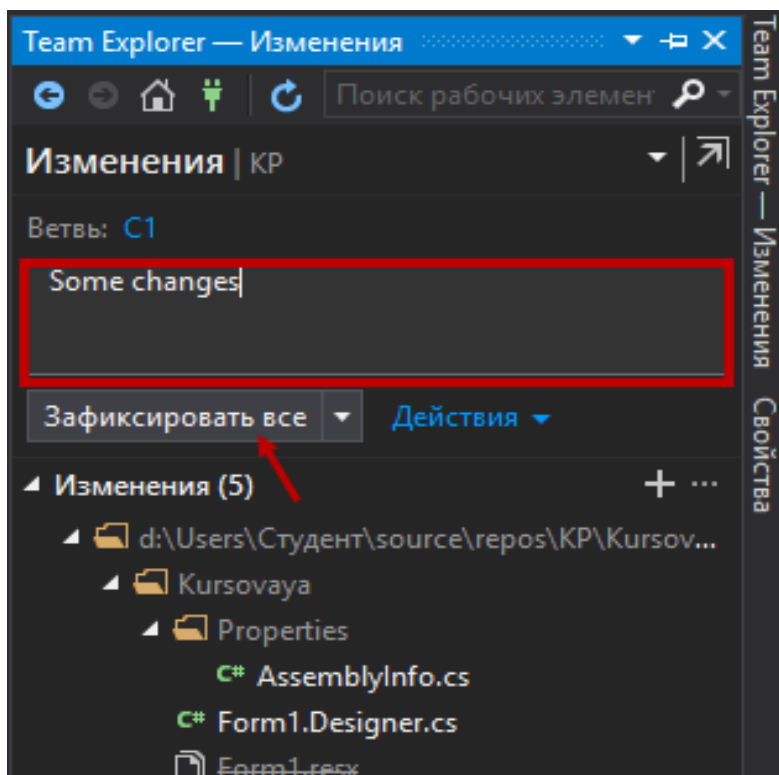
Сначала необходимо зафиксировать изменения в локальном репозитории, затем в удалённом.



Сохранение в локальный репозиторий

Отправка в удаленный репозиторий

# Комментирование изменений и синхронизация



В окне Team Explorer введите обязательный комментарий к изменениям, затем нажмите на кнопку «**Зафиксировать все**» . Для синхронизации с удаленным репозиторием нажмите на ссылку «**Синхронизируйте**»

Нажмите на кнопку «**отправить**»



При успешной синхронизации данных , обе пиктограммы должны показать «0»



# Дополнительные источники

Классный практический курс на русском

<https://githowto.com/ru>

Про remote branches

[Глава в Pro Git](#)

Про reset

<https://git-scm.com/blog/2011/07/11/reset.html>

Основные фишки с картинками на английском

<https://www.atlassian.com/pt/git/tutorial>

Pro Git

<https://git-scm.com/book/ru/v2>

