Git with Gitea

- Git — Основные понятия и как работает
- Git — Установка и первичная настройка

Git — ???????? ??????? ? ??? ????????

Эта страница объясняет базовые принципы работы Git и ключевые термины системы контроля версий. Здесь разобрано, что такое репозиторий, коммит, ветки и как Git хранит историю изменений. Материал рассчитан на начинающих и помогает понять **как устроена модель Git**, что значительно упрощает дальнейшую работу с командами.

??? ????? Git

Git — это распределённая система контроля версий (Version Control System, VCS).

Она позволяет:

- хранить историю изменений проекта
- работать нескольким разработчикам одновременно
- возвращаться к предыдущим версиям кода
- создавать параллельные версии разработки (ветки)

Git используется практически во всех современных проектах разработки.

??? ????? Gitea

Gitea — это сервер для хранения Git-репозиториев.

Он выполняет функции:

- хранения репозиториев
- управления доступом разработчиков
- совместной работы над кодом
- просмотра истории изменений через веб-интерфейс

В нашей инфраструктуре Gitea доступен по адресу:

```
https://git.ext.flamy.studio/
```

Через него происходит:

- хранение проектов
- создание репозиториев
- работа с ветками
- Pull Request и code review

???????? ??????? Git

Чтобы работать с Git, важно понимать несколько базовых терминов.

??????????? (Repository)

Репозиторий — это хранилище проекта и всей истории его изменений.

Он содержит:

- исходный код
- историю изменений
- ветки проекта

Репозиторий может быть:

Тип	Описание
Local repository	копия проекта на компьютере
Remote repository	репозиторий на сервере (Gitea)

Commit

Commit — это сохранённое изменение проекта.

Каждый commit содержит:

- изменения файлов
- автора изменений
- дату
- сообщение коммита

Пример сообщения коммита:

```
Add login validation
```

История проекта состоит из последовательности commit-ов.

Branch (?????)

Branch — это отдельная линия разработки.

Ветки позволяют:

- работать над новыми функциями
- исправлять ошибки
- экспериментировать с кодом

Пример структуры веток:

```
main
 ├─ feature/login
 └─ bugfix/header
```

Remote

Remote — это удалённый репозиторий.

Чаще всего используется remote с именем:

```
origin
```

Пример:

```
origin → https://git.ext.flamy.studio/user/project.git
```

??? Git ?????? ??????????

Git не сохраняет просто файлы — он хранит **снимки состояния проекта**.

Каждый commit — это состояние всех файлов на момент сохранения.

Пример:

```
Commit A → начальное состояние
Commit B → добавлен файл login.js
Commit C → исправлена ошибка
```

Это позволяет:

- смотреть историю изменений
- сравнивать версии
- откатывать проект назад

?????? ?????? Git

Очень важно понимать **как происходит сохранение изменений**.

Git использует три области:

```
Working Directory
        ↓
Staging Area
        ↓
Repository
```

Working Directory

Это файлы проекта на вашем компьютере.

Здесь вы:

- редактируете код
- добавляете файлы
- удаляете файлы

Git отслеживает изменения, но **ещё не сохраняет их в историю**.

Staging Area

Staging area — это промежуточная зона перед commit.

Сюда добавляются изменения, которые войдут в следующий commit.

Команда:

```
git add
```

Repository

Repository — это история commit-ов.

Когда выполняется:

```
git commit
```

изменения сохраняются в истории проекта.

?????????? ????? ?????? Git

```
Редактирование файлов
        ↓
git add
        ↓
Staging Area
        ↓
git commit
        ↓
История репозитория
```

?????? ??????

Предположим, разработчик изменил файл:

```
login.js
```

Git покажет изменения:

```
git status
```

Добавить файл в staging:

```
git add login.js
```

Создать commit:

```
git commit -m "Add login validation"
```

Теперь изменения сохранены в истории проекта.

?????????? ? ?????????? ???????????

Работа Git обычно происходит в двух местах:

```
локальный компьютер
        ↓
Git сервер (Gitea)
```

Процесс работы:

```
edit → commit → push
```

Push

Отправляет изменения на сервер.

```
git push
```

Pull

Получает изменения с сервера.

```
git pull
```

?????? Git ?????? ??? ?????????? ??????

Git позволяет нескольким разработчикам работать над одним проектом одновременно.

Например:

```
Developer A → работает над login
Developer B → работает над profile
```

Каждый разработчик работает в своей ветке, а затем изменения объединяются.

??? ????? ??????????

Git работает по простой модели:

```
изменение файлов
       ↓
git add
       ↓
git commit
       ↓
git push
```

Эта последовательность является **основным рабочим процессом Git**.

??????? ???????

```
Repository  → хранилище проекта
Commit      → сохранённое изменение
Branch      → линия разработки
Remote      → удалённый репозиторий
Working Dir → файлы проекта
Staging     → подготовка commit
```

????

Git хранит историю проекта в виде последовательности commit-ов и позволяет разработчикам безопасно работать с кодом.

Понимание базовой модели Git (Working Directory → Staging → Commit) является основой дальнейшей работы с системой контроля версий.

Git — ????????? ? ????????? ?????????

Эта страница объясняет, как установить Git и выполнить базовую настройку перед началом работы. Здесь разобраны установка Git, настройка имени и email пользователя, проверка конфигурации и настройка хранения учетных данных для работы через HTTPS. Материал рассчитан на начинающих и описывает **настройку Git, которую нужно выполнить один раз перед началом работы**.

????????? Git

Git необходимо установить на компьютер перед началом работы с репозиториями.

Скачать установщик можно с официального сайта:

```
https://git-scm.com/downloads
```

Git доступен для всех основных операционных систем:

- Windows
- Linux
- macOS

???????? ?????????

После установки необходимо проверить, что Git доступен в системе.

```
git --version
```

Пример результата:

```
git version 2.43.0
```

Если команда выполняется успешно — Git установлен правильно.

????????? ????????? Git

Перед началом работы необходимо указать имя пользователя и email.

Эта информация будет использоваться в каждом commit и позволит определить автора изменений.

Настройка выполняется **один раз**.

??????? ??? ????????????

```
git config --global user.name "Ваше имя"
```

Пример:

```
git config --global user.name "Ivan Petrov"
```

??????? email

```
git config --global user.email "email@example.com"
```

Пример:

```
git config --global user.email "ivan.petrov@flamy.studio"
```

Важно использовать **тот же email, который указан в Gitea**.

???????? ????????????

Проверить текущие настройки можно так:

```
git config --list
```

Пример результата:

```
user.name=Ivan Petrov
user.email=ivan.petrov@flamy.studio
```

????????? ?????????? ????????

Имя пользователя:

```
git config user.name
```

Email:

```
git config user.email
```

?????????? ? ????????? ???????????

Git поддерживает два уровня конфигурации.

Тип	Описание
Global	применяется ко всем репозиториям
Local	применяется только к текущему репозиторию

?????????? ???????????

Используется чаще всего.

```
git config --global user.name
```

Настройки сохраняются в файле:

```
~/.gitconfig
```

????????? ???????????

Можно задать настройки только для одного проекта.

```
git config user.name "Developer"
```

Настройки сохраняются в файле:

```
project/.git/config
```

?????? ????? HTTPS

В нашей инфраструктуре Git используется через **HTTPS**.

Пример ссылки репозитория:

```
https://git.ext.flamy.studio/user/project.git
```

Это означает, что Git будет запрашивать:

- логин
- пароль или access token

????????????? Access Token

Вместо пароля рекомендуется использовать **Access Token**.

Это безопаснее и удобнее.

???????? ?????? ? Gitea

1. Открыть **Settings**
2. Перейти в **Applications**
3. Создать **New Access Token**

Обязательно выдайте разрешение на чтение и запись для repository

После создания токена будет показан секретный ключ.

Его нужно сохранить.

????????????? ??????

При первом `git push` Git запросит:

```
Username
Password
```

Вводится:

Поле	Значение
Username	имя пользователя Gitea
Password	Access Token

?????????? ??????? ??????

Чтобы Git не запрашивал пароль при каждом push, можно включить credential helper.

Windows

```
git config --global credential.helper manager
```

---

macOS

```
git config --global credential.helper osxkeychain
```

Linux

```
git config --global credential.helper cache
```

???????? ???????? Git

Полезно проверить общую конфигурацию:

```
git config --global --list
```

???????? ?????? ??? ?????????

???????????? email

Если email отличается от указанного в Gitea, commit может отображаться некорректно.

?? ????????? credentials

Git может запрашивать пароль при каждом push.

Решение — настроить credential helper.

?????????? Git ? ???????

Если команда `git` не работает, нужно проверить установку.

??????? ???????

```
git --version

git config --global user.name "Name"
git config --global user.email "email@example.com"

git config --list
git config --global --list

git config --global credential.helper manager
```

????

Перед началом работы с Git необходимо:

1. установить Git
2. указать имя пользователя
3. указать email
4. настроить хранение учетных данных

Эти действия выполняются один раз и позволяют корректно работать с Git-репозиториями.