

# ВВЕДЕНИЕ

---

Спикер: Фёдор Савчук

О спикере

# Фёдор Савчук

- ◆ Team Lead в ИТ-Холдинг Т1
- ◆ Опыт разработки более 9 лет



# В ЭТОЙ ТЕМЕ

01

Что такое система  
контроля версий

02

Установка Git на Windows,  
Linux и MacOS

03

Ключевые команды  
для работы с файлами

## Определение

# GIT

– это система для контроля версий кода, которая помогает разработчикам следить за изменениями в проектах, работать совместно и хранить историю изменений

# Когда Git становится незаменимым?

## Работа в команде

Git позволяет нескольким разработчикам одновременно работать над одним проектом

## Отслеживание изменений в проекте

легко видеть, кто и когда внес изменения

## Восстановление предыдущих версий

если что-то пошло не так, можно вернуть проект к предыдущему состоянию



# Преимущества Git

## Безопасность

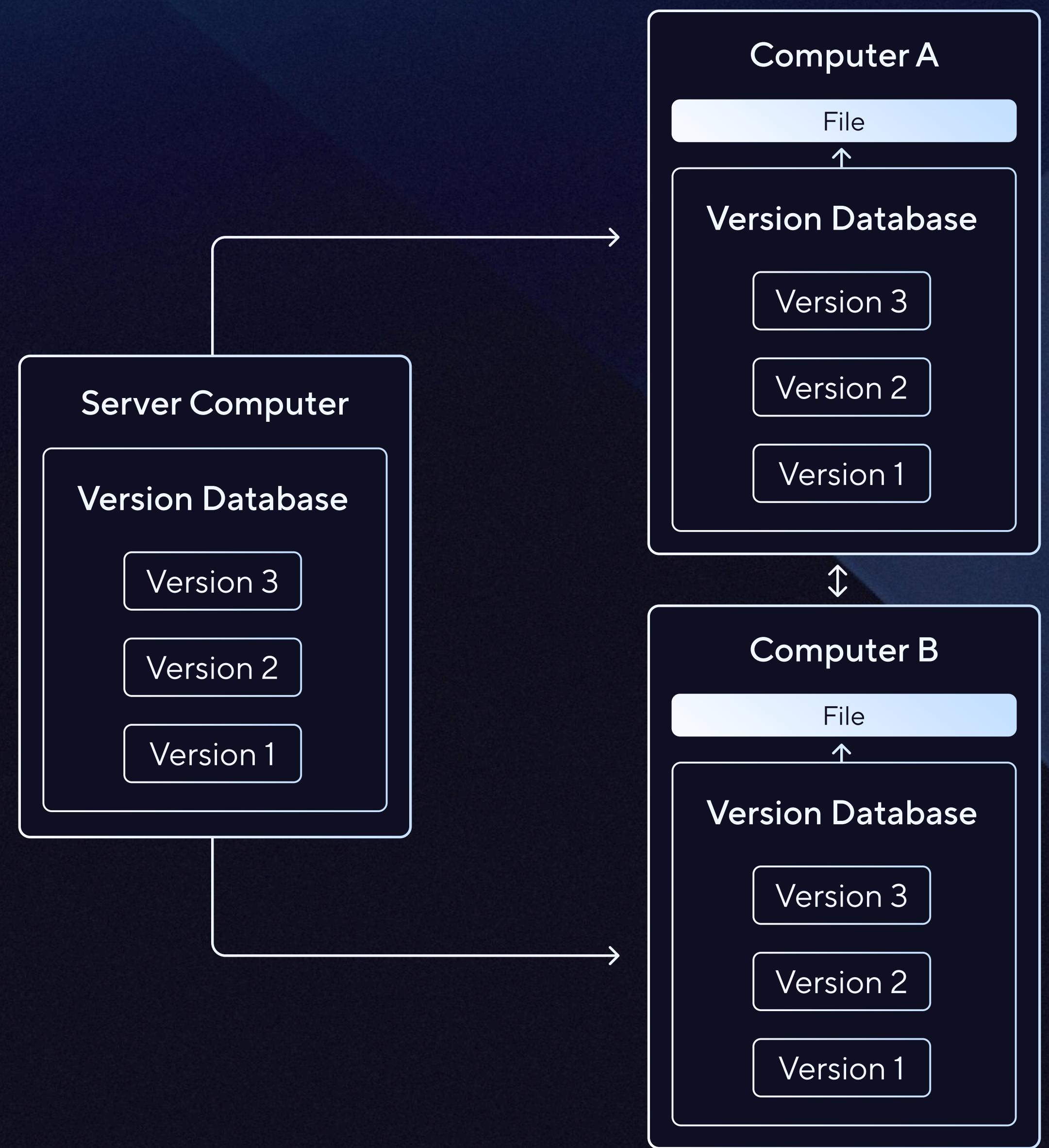
все изменения сохраняются, и вы можете восстановить предыдущие версии

## История изменений

можно просмотреть историю изменений и понять, почему были сделаны те или иные правки

## Совместная работа

упрощает работу над проектами с участием нескольких человек



# Основные понятия Git

## РЕПОЗИТОРИЙ repository

Хранилище, где хранится твой код вместе с историей всех изменений

## КОММИТ commit

Сохраненная версия кода. Каждый раз, когда ты фиксируешь изменения, создаётся новый коммит с уникальным идентификатором

## СЛИЯНИЕ merge

Процесс объединения веток, когда изменения из одной ветки добавляются в другую

## КЛОНИРОВАНИЕ clone

Копирование существующего репозитория на твой компьютер, чтобы начать работу с ним

## ВЕТКА branch

Отдельная линия разработки. Например, можно создать ветку для тестирования новой функции, а потом объединить ее с основной веткой, когда всё будет готово

# ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ GIT

# Примеры использования Git

## Проектирование программного обеспечения

Разработка новых функций и исправление ошибок

## Документация

Отслеживание изменений в документах и инструкциях

## Управление конфигурациями

Хранение настроек и конфигурационных файлов

# УСТАНОВКА GIT НА LINUX

## Установка Git на linux

Для дистрибутивов на базе  
Debian (например, Ubuntu):

```
sudo apt update  
sudo apt install git
```

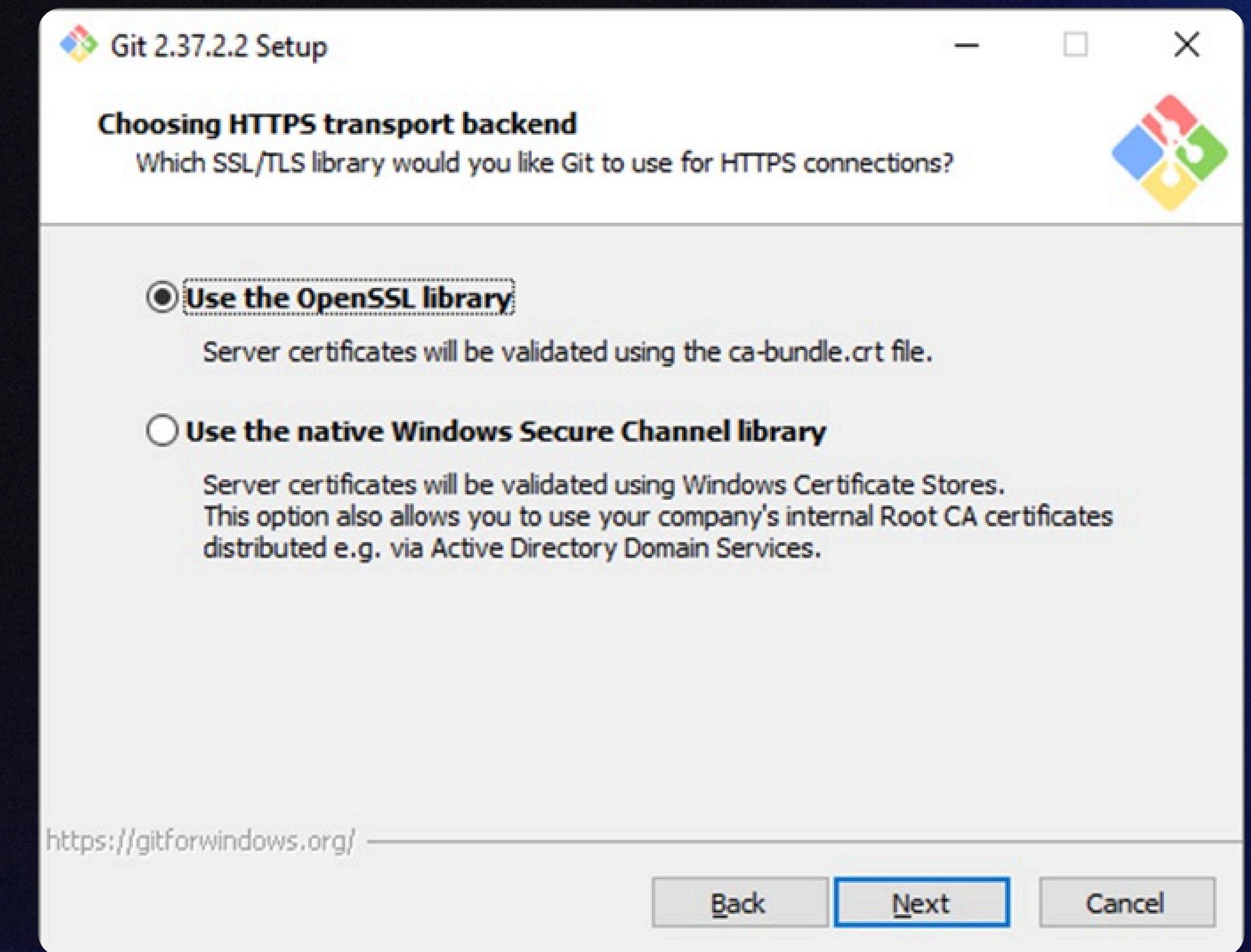
Для Fedora и других  
дистрибутивов на базе RPM:

```
sudo dnf install git-all
```

# УСТАНОВКА GIT НА WINDOWS

## Установка GIT на windows

- 01 Перейдите на официальный сайт Git для Windows и скачайте установочный файл
- 02 Запустите скачанный файл и следуйте инструкциям установщика
- 03 Рекомендуется использовать командную оболочку, входящую в состав msysGit, так как она поддерживает все команды Git



# Настройка Git после установки

После установки необходимо настроить Git, чтобы связать ваши коммиты с вашей идентификационной информацией:

```
git config --global user.name "Ваше Имя"  
git config --global user.email "ваш_email@example.com"
```

Проверьте настройки с помощью команды

```
git config --list
```

# Архитектура Git

Git использует три основных состояния для управления файлами:

01

## Рабочая директория

место, где вы работаете над проектом

02

## Индексация

staging area

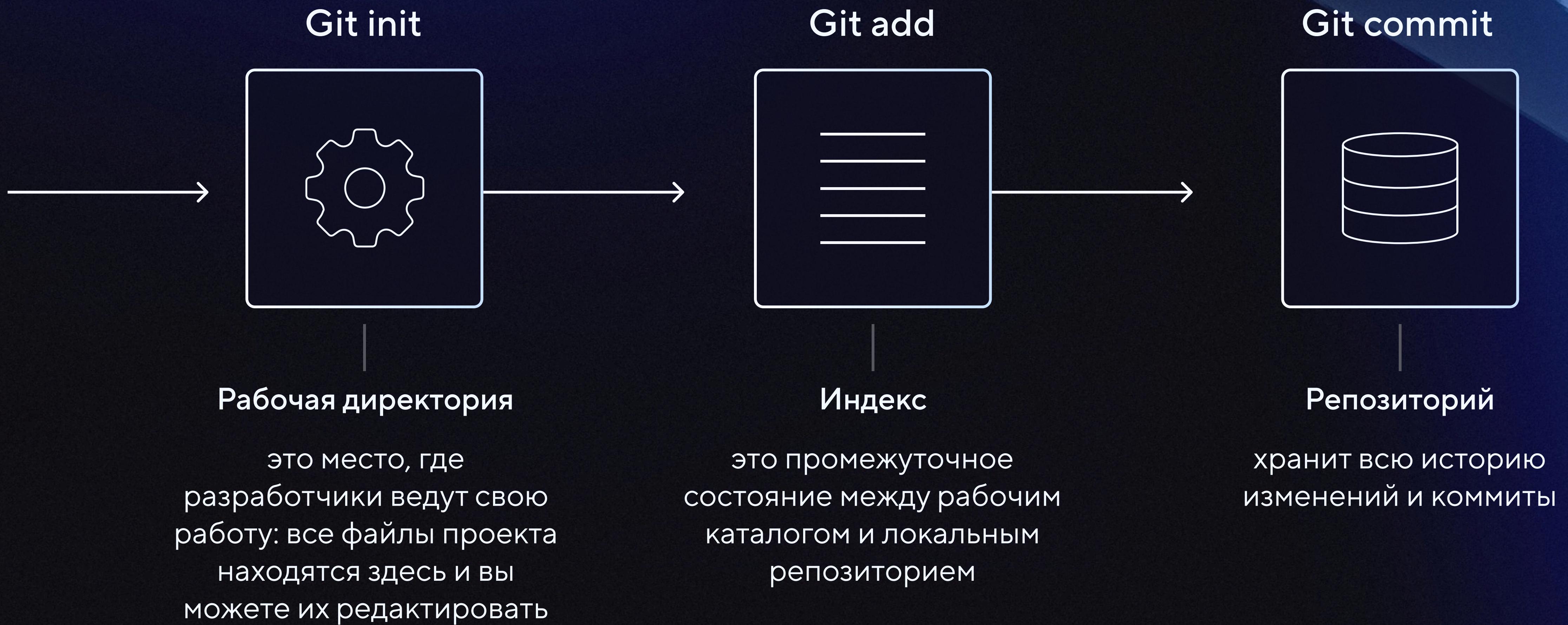
промежуточное состояние, где вы добавляете изменения перед коммитом

03

## Локальный репозиторий

хранит все ваши коммиты и историю изменений

# Основные принципы размещения файлов в git



# Основные состояния файлов

## НЕОТСЛЕЖИВАЕМЫЕ

Untracked

Файлы, которые были добавлены в рабочий каталог, но еще не находятся под контролем версий. Git не отслеживает изменения в этих файлах до тех пор, пока они не будут добавлены в индекс

Пример: новый файл, который вы только что создали

## ОТСЛЕЖИВАЕМЫЕ

Tracked

Файлы, которые находятся под контролем версий. Эти файлы могут быть в следующих подкатегориях

- ♦ **Неизменённые** Unmodified

Файлы, которые не были изменены с момента последнего коммита

- ♦ **Изменённые** Modified

Файлы, которые были изменены с момента последнего коммита, но еще не добавлены в индекс

- ♦ **Индексированные** Staged

Файлы, которые были добавлены в индекс с помощью команды `git add` и готовы к коммиту

# Переходы между состояниями

## НЕОТСЛЕЖИВАЕМЫЙ → ОТСЛЕЖИВАЕМЫЙ

Чтобы перевести файл из состояния "неотслеживаемый" в "отслеживаемый", используйте команду

```
git add <имя_файла>
```

## ОТСЛЕЖИВАЕМЫЙ (ИЗМЕНЁННЫЙ) → ИНДЕКСИРОВАННЫЙ

Чтобы перевести файл из состояния "изменённый" в "индексированный", используйте команду

```
git add <имя_файла>
```

## ИНДЕКСИРОВАННЫЙ → ЛОКАЛЬНЫЙ РЕПОЗИТОРИЙ

Чтобы зафиксировать изменения и перевести файл из состояния "индексированный" в "локальный репозиторий", используйте команду

```
git commit -m "Описание изменений"
```

# Шаги для добавления файлов в индекс для вашего первого коммита

```
git add
```

Добавление файла в индекс  
для будущего коммита

```
git commit -m "Описание"
```

Коммит с  
описанием

# Основные команды Git

`git init`

Создает новый локальный репозиторий

`git add`

Добавляет изменения в индекс для следующего коммита

`git commit`

Сохраняет изменения в локальном репозитории с сообщением

`git push`

Отправляет изменения из локального репозитория в удалённый

`git pull`

Загружает изменения из удалённого репозитория в локальный

# Команды для отслеживания фиксации

`git status`

проверка состояния отслеживаемых  
и неотслеживаемых файлов

`git diff`

просмотр различий между рабочей  
директорией и индексом

`git log`

просмотр истории  
коммитов

# ВЫВОДЫ

01

**Git** – необходимый инструмент для разработки программного обеспечения. Он обеспечивают надежность, безопасность и удобство работы как для индивидуальных разработчиков, так и для команд

02

Git предоставляет мощные инструменты для управления версиями кода, позволяя разработчикам эффективно работать как индивидуально, так и в команде. Понимание принципов работы Git – ключ к успешному управлению проектами

03

**Установка Git** – это первый шаг к эффективному управлению версиями вашего кода. Правильная настройка после установки позволяет вам начать работу с системой контроля версий. Понимание процесса установки и настройки Git является важным для успешной работы в команде и управления проектами