СИНЕРГИЯ АКАДЕМИЯ

GIT

GITHUB И GITLAB

Спикер: Фёдор Савчук

В ЭТОЙ ТЕМЕ

01
Установка SSH-агента
и использование ключей

02
Ключевые команды для работы
с удаленными репозиториями

03
GitHub как социальная сеть
и часть портфолио

04
GitLab и дополнительные
возможности

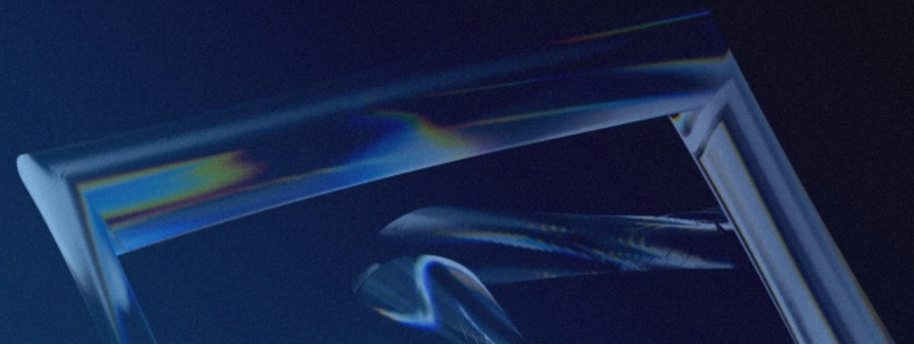
Определение

УДАЛЁННЫЕ РЕПОЗИТОРИИ

это версии вашего проекта, которые находится уже не локально на вашем устройстве, а на сервере. Они обеспечивают доступ к вашему коду другим пользователям и устройствам

Зачем они нужны?

- Удалённые репозитории помогают в сотрудничестве над проектом, резервировании данных и обеспечивают возможность непрерывной интеграции
- Они позволяют легко делиться кодом и управлять им в команде



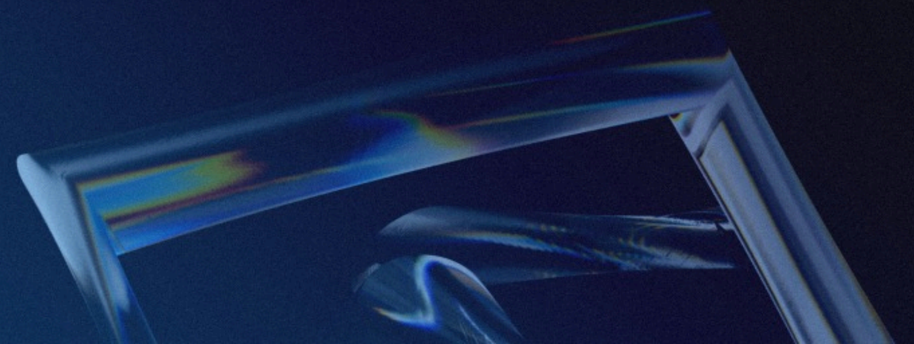
Основные команды

Существует несколько ключевых команд для работы с удалёнными репозиториями:

`git remote`	позволяет управлять списком удалённых репозиториев
`git fetch`	получает изменения из удалённого репозитория, не сливая их с вашей локальной копией
`git pull`	подтягивает изменения и сливает их с вашей локальной копией
`git push`	отправляет ваши изменения в удалённый репозиторий

Git Remote

Команда git remote позволяет просматривать и управлять подключениями. Например, git remote add origin <URL> добавляет новый удалённый репозиторий с именем "origin"



Git Fetch

Git Pull

Хотя git fetch и git pull могут казаться схожими, они служат разным целям. git fetch только загружает изменения, а git pull также пытается автоматически объединить их с вашей текущей веткой

Определение

ПУБЛИКАЦИЯ В УДАЛЁННЫЙ РЕПОЗИТОРИЙ

это важный этап работы с Git, который позволяет делиться изменениями с коллегами и интегрировать их в общий проект. Этот процесс обеспечивает совместную разработку и помогает работать над проектами в команде

Команды для публикации

`git push`

- **Описание:** Отправляет ваши локальные изменения на удалённый репозиторий
- **Синтаксис:** git push [remote] [branch]
- **Пример:** git push origin main

`git pull`

- **Описание:** Загружает изменения из удалённого репозитория и объединяет их с локальными
- **Синтаксис:** git pull [remote] [branch]
- **Пример:** git pull origin main

`git clone`

- **Описание:** Клонирует удалённый репозиторий на ваш локальный компьютер
- **Синтаксис:** git clone [url]
- **Пример:** git clone https://github.com/user/repo.git

Распространенные ошибки

Конфликты при слиянии

Возникают, если изменения от разных участников затрагивают одну и ту же часть кода

Ошибка доступа

Отсутствие прав на запись в удалённый репозиторий

Несинхронизированная история коммитов

Может потребовать использования команды git fetch перед git pull

GitHub

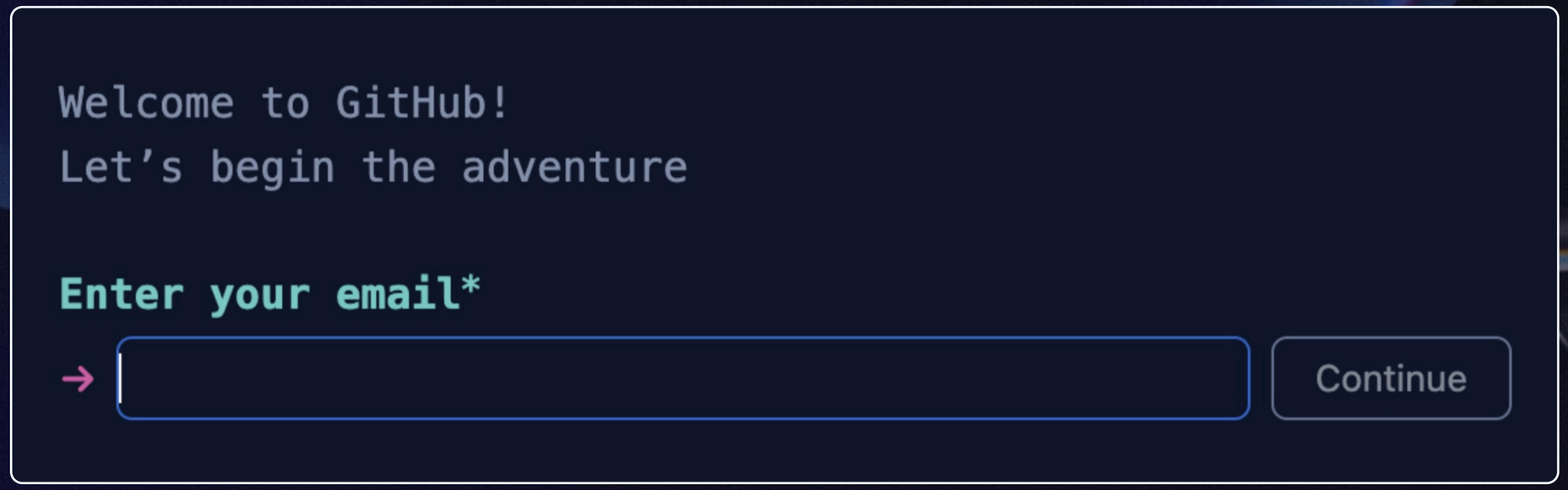
это популярная платформа для управления Git-репозиториями и совместной работы. Она предоставляет пользователям возможность хранить и делиться проектами

Регистрация на Github

01 Перейдите на сайт github.com (https://github.com/)

02 Нажмите "Sign up"

03 Заполните форму регистрации



Настройка учётной записи git

Команды

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

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

Эти команды задают имя и email, которые будут использоваться при коммитах

Создание первого репозитория

01 На главной странице GitHub нажмите "New" для создания нового репозитория

02 Задайте параметры: имя репозитория, описание, настройка приватности

ВВЕДЕНИЕ В SSH

Определение

SECURE SHELL
SSH

это криптографический сетевой протокол для безопасного оперирования над сетевыми службами. В GitHub SSH используется для аутентификации и безопасного подключения к репозиторию без необходимости вводить логин и пароль. Это важный этап в обеспечении безопасности доступа к вашему коду

Установка ssh-агента

SSH-АГЕНТ

это программа, которая управляет вашими SSH-ключами. Она работает в фоновом режиме и позволяет вам использовать ваши ключи без необходимости вводить пароль каждый раз. Windows 10 и выше имеют встроенный SSH-клиент

Для установки SSH-агента на Unix-системах (Linux, macOS) используйте команду

```
eval "$(ssh-agent -s)"
```

Генерация ssh-ключей

Создание SSH-ключей осуществляется командой

```
ssh-keygen -t rsa -b 4096 -C "ваш.email@пример.com
```

Эта команда создает пару ключей (открытый и закрытый) с высокой степенью шифрования. Обязательно сохраните закрытый ключ в надёжном месте

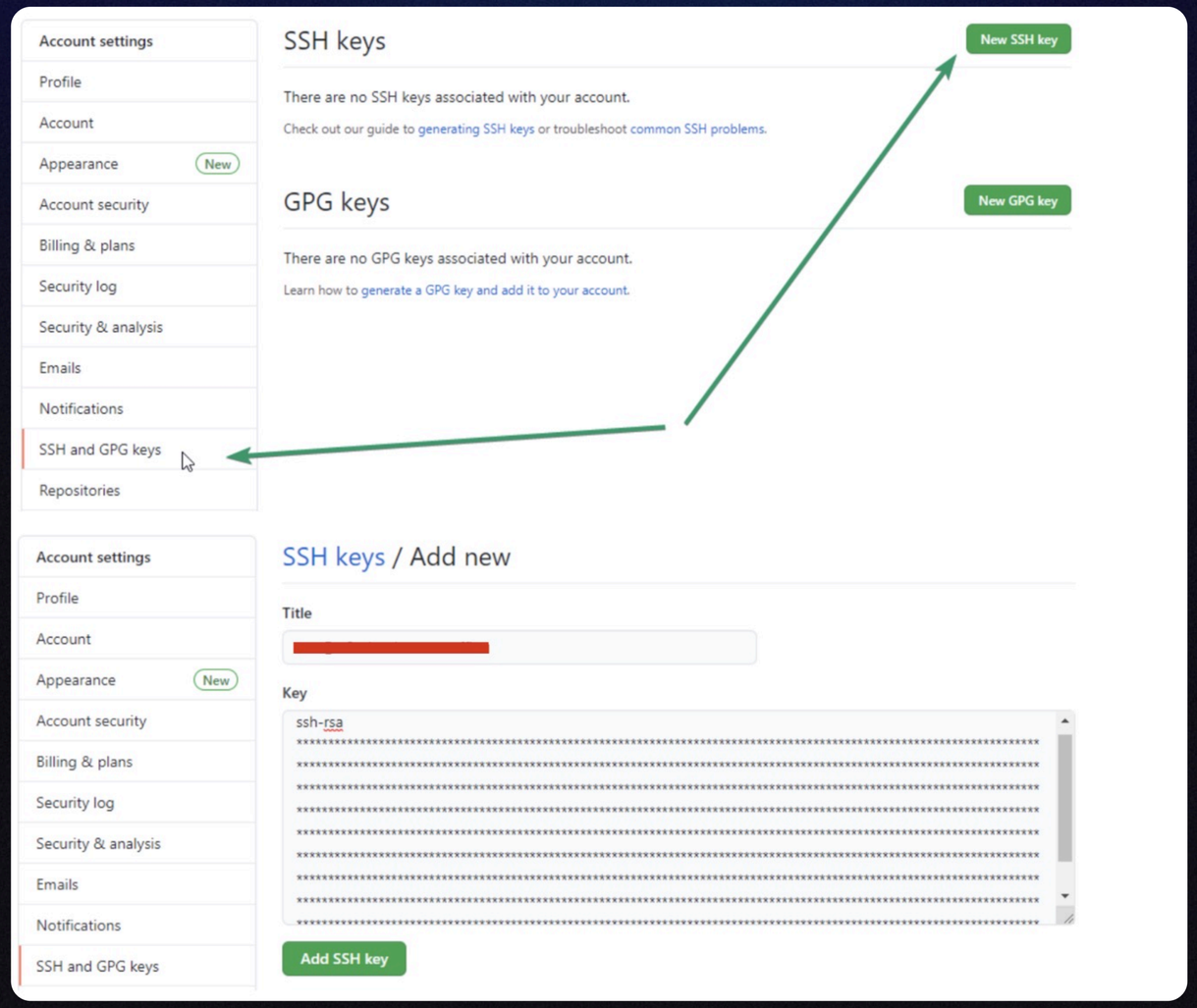
Добавление ssh-ключа в GitHub

Для доступа к GitHub используйте ваш открытый ключ. Скопируйте его:

```
cat ~/.ssh/id_rsa.pub
```

Затем, в настройках вашего профиля GitHub, добавьте его в раздел "SSH and GPG keys"

Добавление ssh-ключа в GitHub



Использование ssh для git-операций

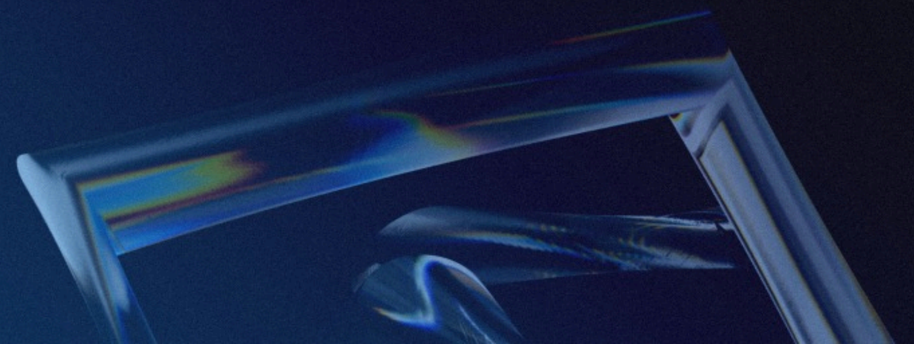
После настройки SSH, операции с Git (например, `git clone`, `git push`, `git pull`) автоматически будут использовать SSH-ключи для аутентификации и не потребуют ввода пароля

Git

Участие в разработке чужих проектов на GitHub позволяет расширить свои навыки, внести вклад в open-source и завести полезные профессиональные контакты

Fork и Pull Request

Основной процесс участия в чужом проекте включает в себя создание своей копии репозитория (fork), внесение изменений и предложение изменений основному проекту через Pull Request



Fork

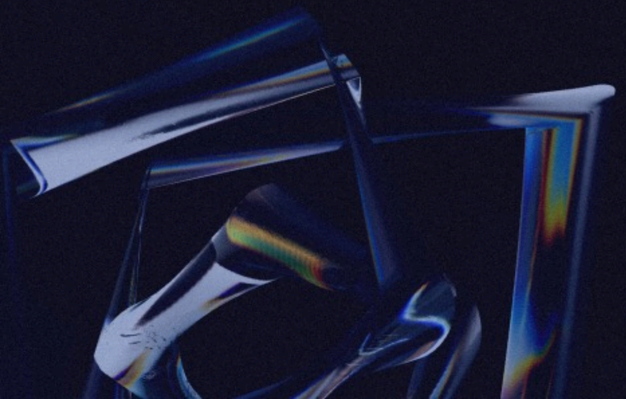
Используется в интерфейсе GitHub с кнопкой "Fork".
Копирует проект в ваш аккаунт для работы над ним

Pull Request

01 Создайте новую ветку для своих изменений

02 Выполните коммит изменений

03 Откройте Pull Request через интерфейс GitHub



GitHub

позволяет следить за обновлениями, ставить звезды проектам, комментировать и создавать форки. Эти функции помогают быть в курсе трендов и общаться с коллегами

GitHub как социальная сеть

СОЗДАНИЕ ПРОФИЛЯ

Загрузите фотографию и заполните информацию о себе. Добавьте README в свой профиль с кратким обзором вашего опыта и интересов

ВЗАИМОДЕЙСТВИЕ С ДРУГИМИ ПОЛЬЗОВАТЕЛЯМИ

Участвуйте в обсуждениях, предлагайте улучшения существующих проектов через "pull requests"

ВНЕСЕНИЕ ИЗМЕНЕНИЙ В ОТКРЫТЫЕ ПРОЕКТЫ

Используйте "fork" и "pull requests" для взаимодействия с сообществом. Это может улучшить ваши навыки и укрепить профессиональные связи

СТРАТЕГИИ ПОСТРОЕНИЯ ПОРТФОЛИО

Публикуйте собственные проекты, участвуйте в репозиториях других пользователей, отслеживайте популярные тренды и технологии

GitHub Projects

позволяет следить за обновлениями, ставить звезды проектам, комментировать и создавать форки. Эти функции помогают быть в курсе трендов и общаться с коллегами

Организация задач с помощью issues

Issues в GitHub используются для отслеживания ошибок, обсуждения новых функций и управления другими задачами. Они могут быть связаны с проектами для более детальной организации работы

Управление и отслеживание прогресса

С помощью Projects и Issues можно увидеть общий прогресс команды и быстро определить, какие задачи требуют внимания. Используйте метки и фильтры для эффективного поиска информации.

Определение

GitLab

это веб-платформа DevOps, которая предоставляет репозитории Git, систему отслеживания ошибок и непрерывную интеграцию и доставку (CI/CD)

Отличия GitLab от GitHub

- **GitLab** включает встроенные функции CI/CD, тогда как **GitHub** требует интеграции с внешними сервисами
- **GitLab** предлагает больше возможностей на бесплатной версии, включая приватные репозитории
- Развитие **GitLab** фокусируется на предоставлении инструментов для DevOps полного цикла

Основные функции GitLab для работы с репозиториями

- Создание и клонирование репозиториев
- Управление ветками и коммитами
- Системы контроля доступов и разрешений в репозиториях
- Интеграция с CI/CD для автоматизации процессов

Инструменты контроля версий и совместной работы

Поддержка pull запросов и code-review для улучшения качества кода

Возврат к предыдущим версиям репозитория

Защита веток и механизм разрешения конфликтов при слиянии

Управление CI/CD пайплайнами

- Continuous Integration и Continuous Deployment (CI/CD) – ключевой компонент GitLab для автоматизации тестирования и развертывания
- GitLab CI/CD позволяет настраивать автоматизированные процессы, минимизируя вмешательство человека

Встроенные инструменты обеспечения безопасности

- GitLab включает сканирование на уязвимости, секреты и вредоносный код
- Встроенные отчеты по безопасности помогают разработчикам вовремя устранять проблемы

Возможности мониторинга и аналитики

- GitLab предоставляет инструменты для мониторинга производительности приложений и анализа процессов разработки
- Используйте встроенные дашборды для отслеживания метрик в реальном времени

Платформа для совместной работы и управления проектами

GitLab

это не просто хостинг для репозиториев,
но и платформа для совместной работы

Используйте инструменты для управления проектами,
такие как задачи, доски Kanban и мильстоуны

ВЫВОДЫ

01

SSH-ключи упрощают аутентификацию, позволяя разработчикам безопасно подключаться к репозиториям без необходимости постоянно вводить логин и пароль

02

Знание ключевых команд для работы с удаленными репозиториями, таких как git push, git pull, git fetch и git remote, является необходимым для эффективного управления кодом в команде

03

GitHub не только служит инструментом для управления версиями, но и является важной социальной платформой для разработчиков. Участие в проектах, создание pull requests и взаимодействие с сообществом помогают налаживать профессиональные связи и развивать навыки

04

GitLab предлагает дополнительные функции, такие как интегрированные инструменты для непрерывной интеграции и доставки (CI/CD), управление проектами и возможности для автоматизации рабочего процесса