

Тестирование ПО

Системы контроля версий

Кулаков Кирилл Александрович

Проблема обмена данными

- Сложный проект -> большая команда -> много кусочков кода
- Использование традиционных способов обмена данными:
 - Сетевые папки
 - Дискеты / Компакт диски / Флешки
 - Использование соц. сетей для передачи файлов
 - Облачные папки (Яндекс.диск)
 - Архивы на файлообменниках
 - Электронная почта

Системы контроля версий

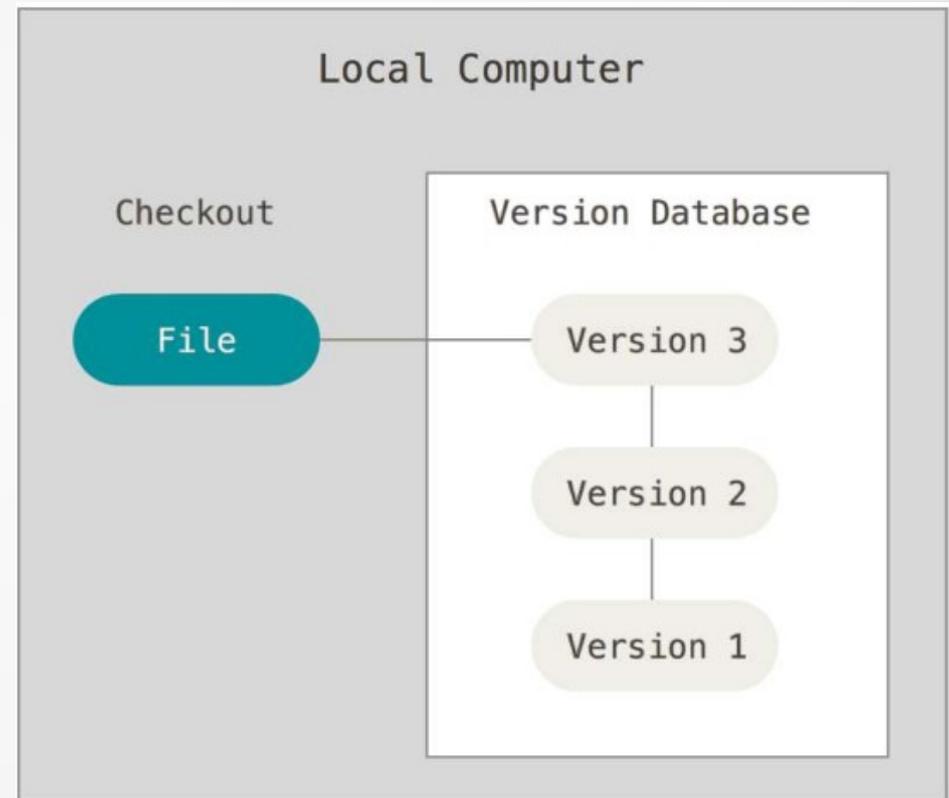
- **Контроль версий** (англ. Version control), также известный как контроль исходного кода (англ. Source control) - это практика отслеживания и управления изменениями в программном коде.
- **Системы контроля версий** (англ. version control systems или VCS, часто встречается название Revision Control System или RCS) - это программные инструменты, которые помогают группам разработчиков управлять изменениями исходного кода с учетом хронологии.
- Главные возможности системы контроля версий:
 - Полная история изменений каждого файла за длительный период
 - Возможность отслеживать каждое изменение, внесенное в файлы
 - Описание причин всех производимых изменений
 - Откат изменений, если что-то пошло не так
 - Совместная работа группы над одним проектом

Системы контроля версий

- **Репозиторий** — место, где хранятся и поддерживаются какие-либо данные.
- Также под репозиторием понимается каталог файловой системы, в котором могут находиться:
 - сами контролируемые файлы
 - журналы конфигураций сборки проекта
 - журнал операций, выполняемых над репозиторием
- **Версией** или **ревизией** (revision) называется конкретное зафиксированное состояние репозитория
- **Управление исходным кодом** (source control management, SCM) используется для отслеживания изменений в репозитории исходного кода

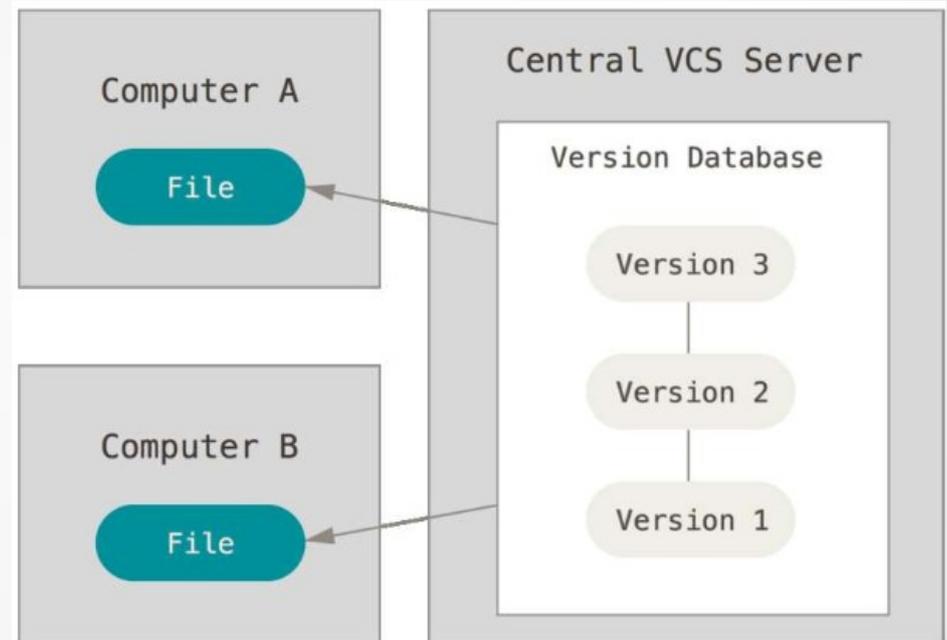
Виды систем контроля версий

- **Локальные системы контроля версий**
- Одной из наиболее известных VCS такого типа является rcs (Revision Control System).
- Утилита основана на работе с наборами патчей между парами версий (патч — файл, описывающий различие между файлами), которые хранятся в специальном формате на диске.
- Она позволяет пересоздать любой файл на любой момент времени, последовательно накладывая патчи.



Виды систем контроля версий

- **Централизованные системы контроля версий**
- Одно основное хранилище всего проекта
- Каждый пользователь копирует себе необходимые файлы из репозитория, изменяет и добавляет свои изменения обратно

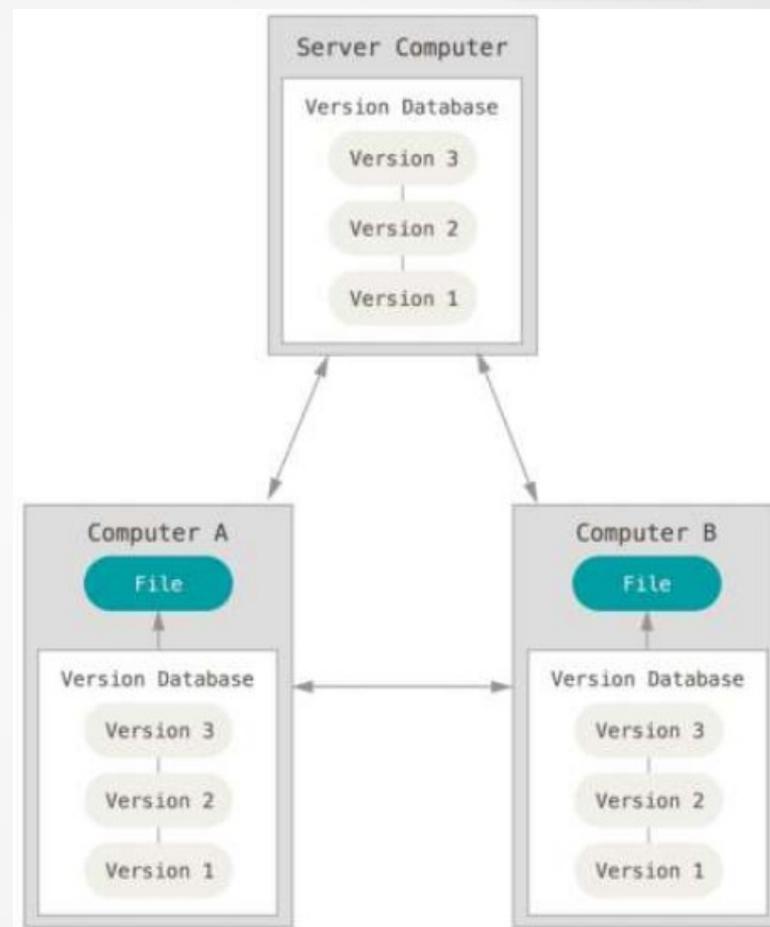


Виды систем контроля версий

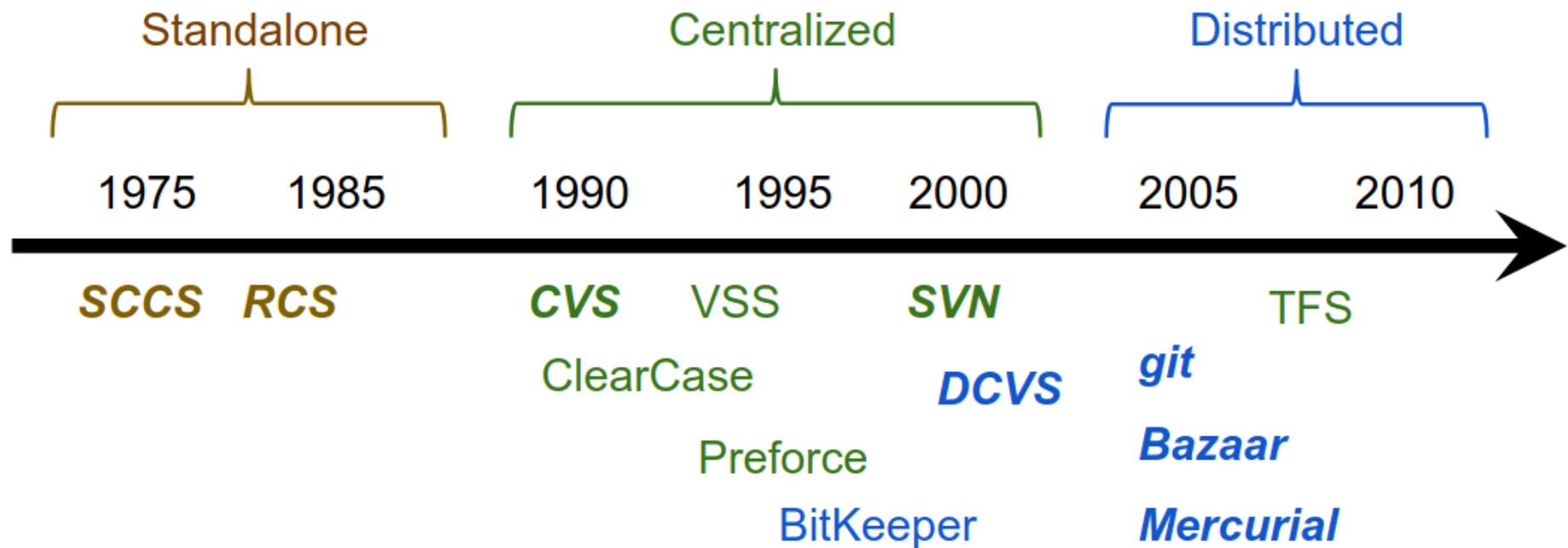
- **Централизованные системы контроля версий**
- Достоинства:
 - Всегда есть выделенная "эталонная" версия файлов
 - Централизованное администрирование
 - Привычный workflow
- Недостатки:
 - Единая точка отказа — сервер
 - Любые изменения влияют на всех пользователей
 - Требуется достаточно гибкая система прав
 - В очень больших или распределенных проектах централизованные механизмы работают плохо.
 - Репозитории уровня корпорации трудно администрировать

Виды систем контроля версий

- **Распределённые системы контроля версий**
- В таких системах как Git, Mercurial клиенты не просто выгружают последние версии файлов, а полностью копируют репозиторий.
- У каждого пользователя свой вариант (возможно не один) репозитория
- При этом можно выделить центральный репозиторий, в который будут отправляться изменения из локальных и с ним же эти локальные репозитории будут синхронизироваться
- Присутствует возможность добавлять и забирать изменения из любого репозитория



История систем контроля версий



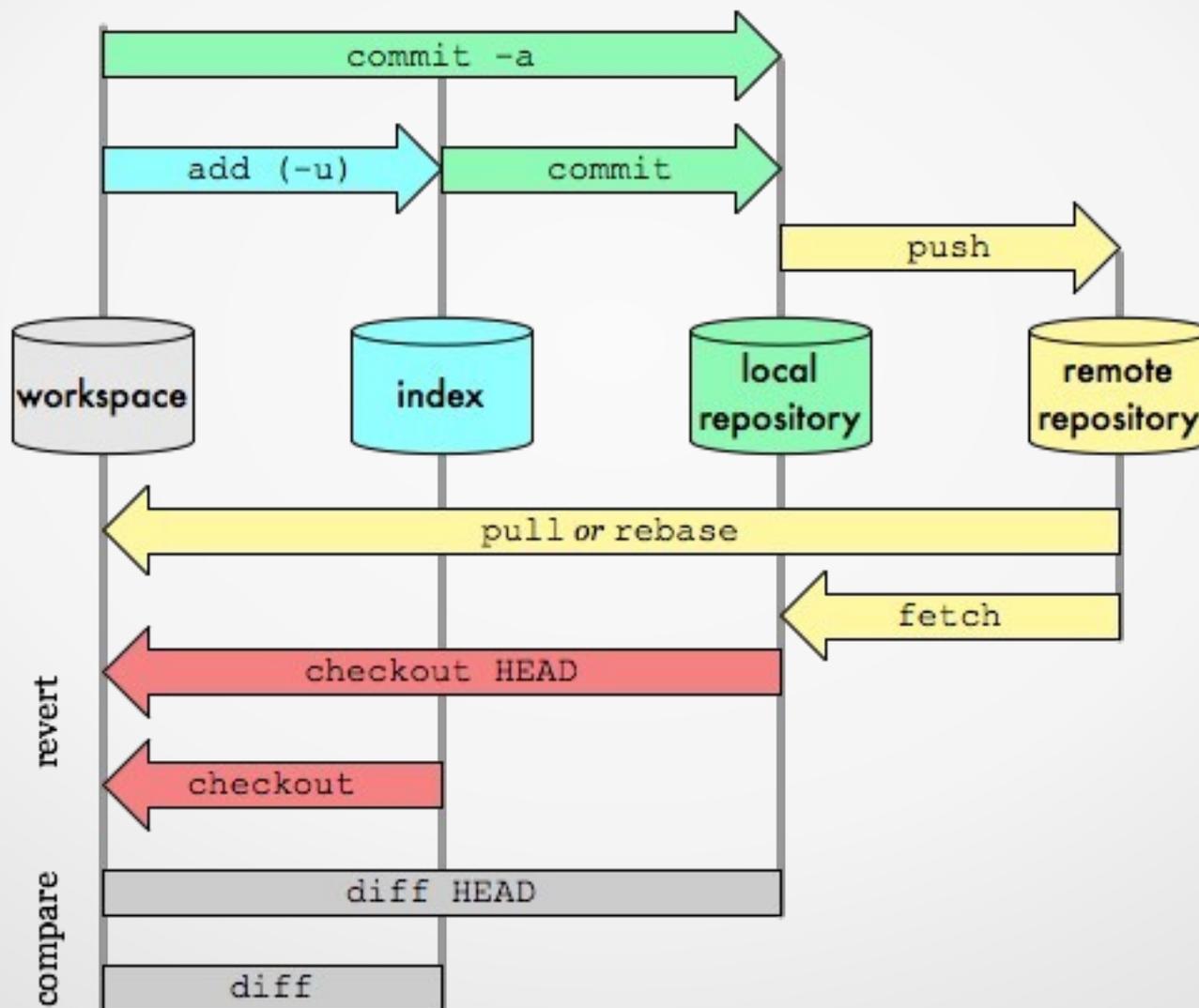
Терминология

- **tag** (Метка) - именованная версия проекта
- **branch** (Ветка) - метка, имеющая историю развития
- **commit** (Коммит, фиксация) - сохранение изменений в репозитории
- **checkout** (Чекаут, извлечение) - получение рабочей копии (без истории)
- **push/pull** — отправка/получение изменений в/из другого репозитория
- **main branch/master branch** - основная ветка
- **Working directory/working copy** (рабочая копия) - снимок одной из версий проекта

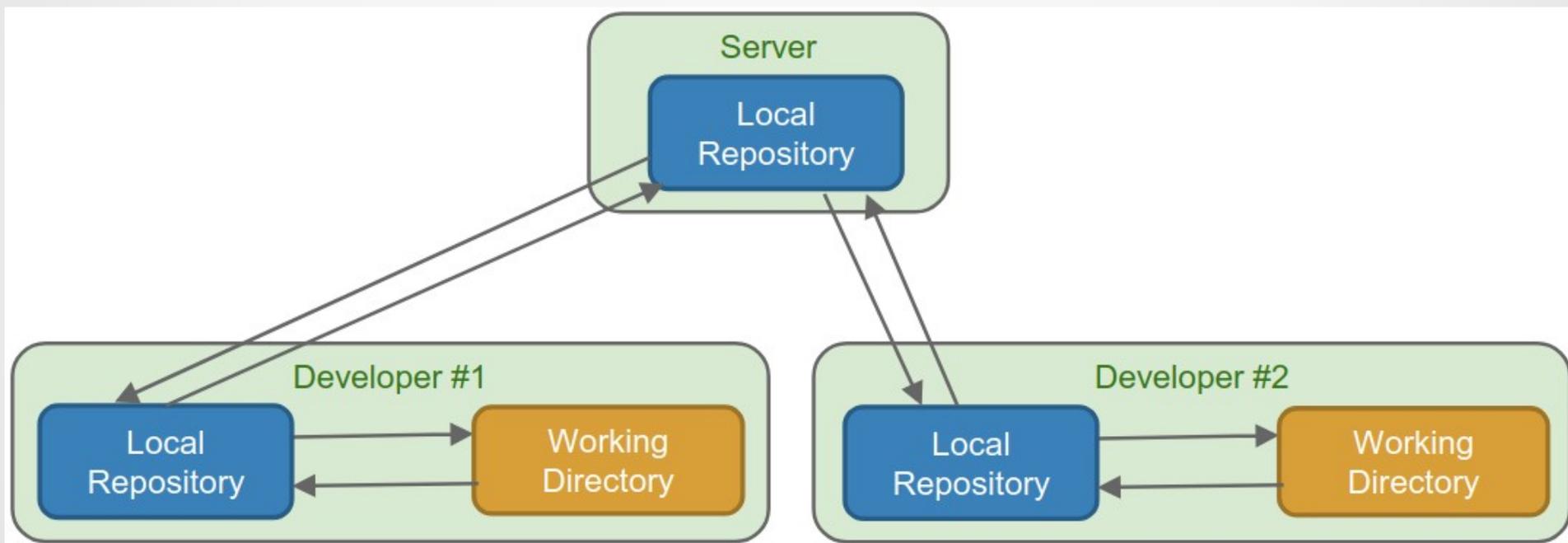
Git: Общая схема работы

Git Data Transport Commands

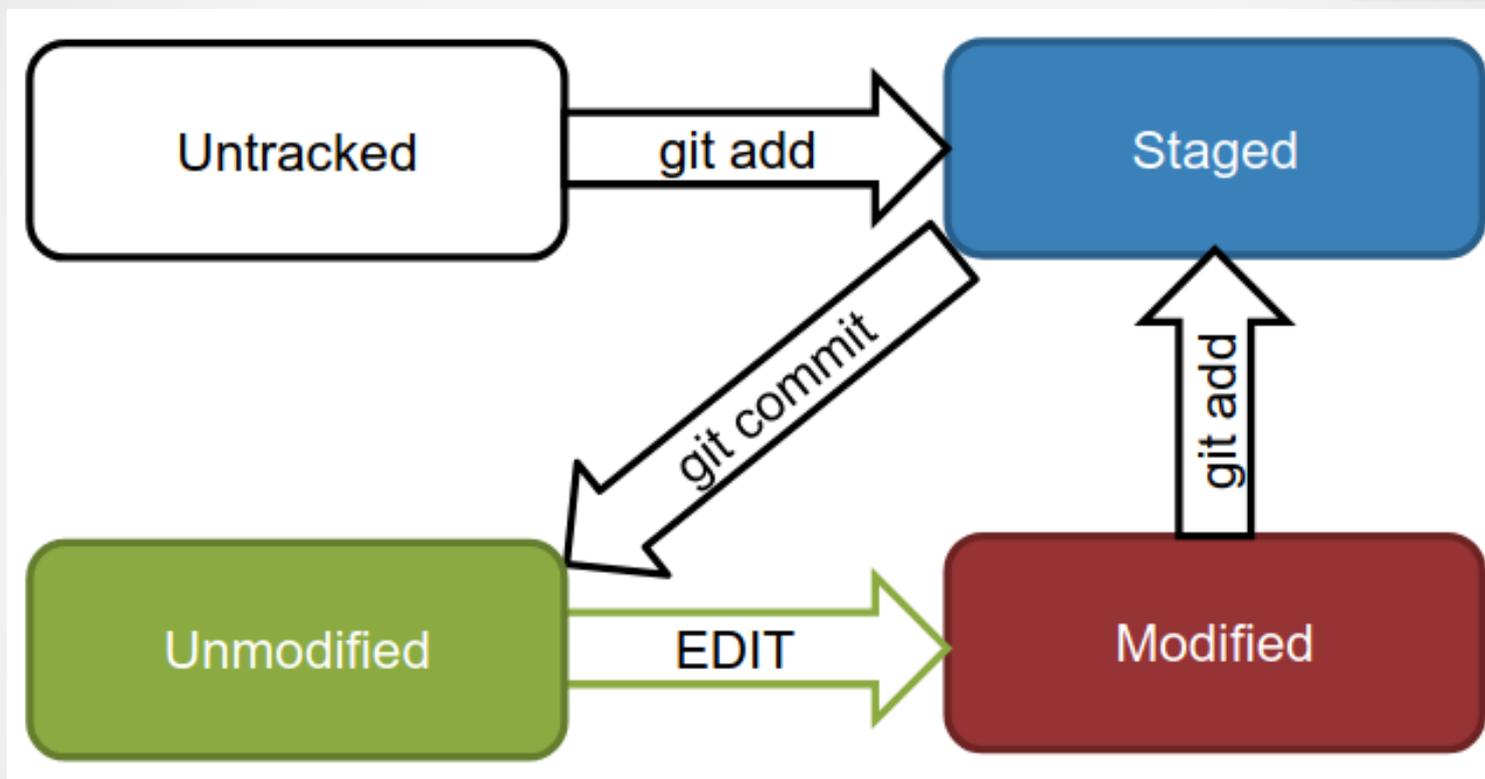
<http://osteele.com>



Git: работа с сервером



Git: Статус файла



Git: Работа с индексом (stage)

- Команда **add** добавляет измененные файлы в stage.
`$git add filename`
- Команда **rm** помечает файл в stage как удаленный.
`$git rm filename`
- Команда **mv** перемещает/переименовывает файл
`$git mv old_filename new_filename`
- Команда **reset** сбрасывает текущий stage.
`$git reset HEAD filename`
- Команда **status** отображает текущий статус индекса (измененные и неотслеживаемые файлы)
- `$git status`

Git: Работа с локальным репозиторием

- Команда **commit** сохраняет текущий stage в локальный репозиторий.

```
$git commit
```

```
$git commit --amend
```

- Команда **branch** создает/удаляет новую ветку.

```
$git branch
```

```
* main
```

```
$git branch test
```

```
$git branch
```

```
* main
```

```
test
```

- Команда **checkout** переключает рабочую копию на другую ветку.

```
$git checkout test
```

```
Switched to branch 'test'
```

Git: Commit message

- Комментарий не должен быть пустым.
- Комментарий должен отвечать на вопрос что и зачем было сделано.
- Комментарий должен быть лаконичным (twitter style)
- Комментарий должен быть на английском языке
- В комментарии указывают ссылку на задачу в багтрекере (+ операцию над ней)

Пример: «Closes #4, #6, Related to #5»

Git: Работа с удаленным репозиторием

- Команда **clone** клонирует репозиторий и создает рабочую копию.

```
$git clone git@github.com:seekerk/gtest.git
```

```
Cloning into gtest...
```

...

- Команда **push** отправляет изменения в удаленный репозиторий.

```
$git push origin main
```

- Команда **pull** забирает изменения указанной ветки из удаленного репозитория и сливает их в текущую ветку.

```
$git pull origin master
```

- Команда **fetch** забирает все изменения из удаленного репозитория.

```
$git fetch
```

Git: Временное хранилище (stash)

- Команда **stash** (или продвинутый вариант **stash push**) помещает изменения из stage во временное хранилище и сбрасывает рабочую копию.

```
$git stash
```

- Команда **stash pop** получает изменения из временного хранилища

```
$git stash pop
```

- Получение изменений без удаления из хранилища с помощью команды **stash apply**

```
$git stash apply
```

- Список спрятанных изменений: **stash list**
- Просмотр спрятанного изменения: **stash show**
- Удаление спрятанного изменения: **stash drop**

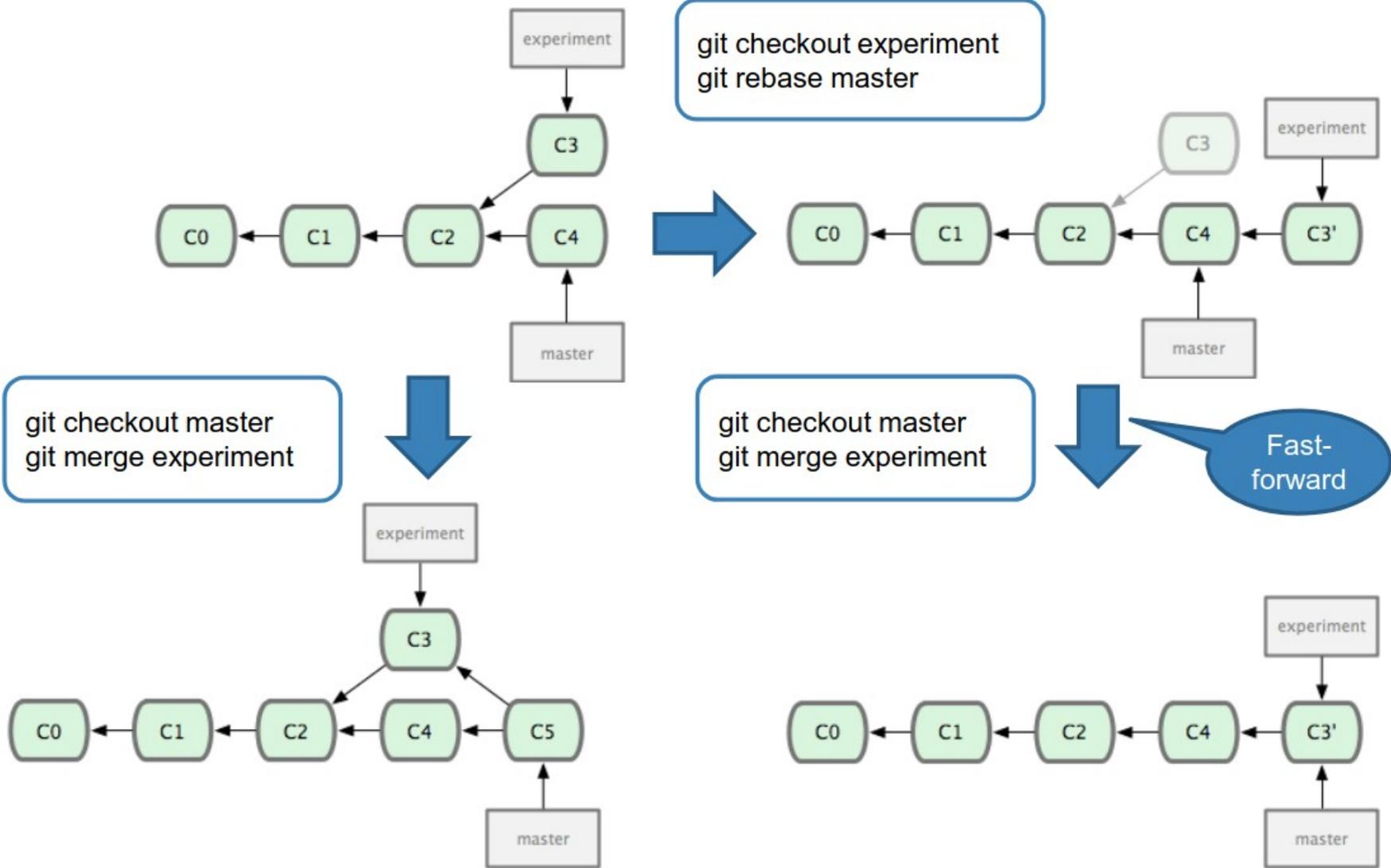
Git: Ключи

- Контентно-адресуемое key-value хранилище (.git)
- В качестве ключа используется SHA-1 хэш от содержимого
- Указатели на выделенные точки в репозитории
- В качестве идентификаторов ревизий используются хэши
 - Общепринято использовать первые 6-8 символов хэша, но это не однозначно

Git: Ветки

- Ветки это указатели на определенные коммиты
- Release branch. Параллельная разработка и поддержка старых версий.
- Feature branch. Разработка нового функционала без риска сломать работающий код.
- Частые коммиты в собственную ветку
- **git branch + git checkout**: создание ветки и переключение на нее
- **git merge**: слияние в текущую ветку
- **git rebase**: перестройка дерева коммитов в текущей ветке

Git: Merge vs. Rebase



Git: Конфликты слияний

- Совместное редактирование одних и тех же строк
- Редактирование рядом стоящих строк
- Изменение бинарных файлов

```
    <div id="navigation">
      <ul>
<<<<<<< HEAD
          <li><a href="index.html">Home</a></li>
          <li><a href="about.html">About Us</a></li>
          <li><a href="product.html">Product</a></li>
          <li><a href="imprint.html">Imprint</a></li>
=====
          <li><a href="returns.html">Returns</a></li>
          <li><a href="faq.html">FAQ</a></li>
>>>>>> develop
      </ul>
    </div>
```

Git: Конфликты слияний

Local Changes (Read-only)	LF	Result	CRLF	Changes from Server (revision 50314cc8638a07fa4a88653d0752ec656f9... LF)
package ca.cmpt276.mergedemo;	1	1	package ca.cmpt276.mergedemo;	1
/**	2	2	/**	2
* NumberFun class manages some data an genera	3	3	* NumberFun class manages some data an generates	3
* My Changes applied!	4	4	* Base version	4
*/	5	5	*/	5
public class NumberFun {	6	6	public class NumberFun {	6
private int[] data;	7	7	private int[] data;	7
public NumberFun(int[] data) {	8	8	public NumberFun(int[] data) {	8
this.data = data;	9	9	this.data = data;	9
}	10	10	}	10
public int getMin() {	11	11	public int getAverage() {	11
int min = data[0];	12	12	int sum = 0;	12
for (int value : data) {	13	13	int i = 0;	13
if (value < min) {	14	14	while (i < data.length) {	14
min = value;	15	15	sum += data[i];	15
}	16	16	i++;	16
}	17	17	}	17
return min;	18	18	return sum / data.length;	18
}	19	19	}	19
public int getAverage() {	20	20	public void printData() {	20
if (data.length == 0) {	21	21	// Print out the data	21
return 0;	22	22	int i = 0;	22
}	23	23	while (i < data.length) {	23
int sum = 0;	24	24	int value = data[i];	24
int i = 0;	25	25	System.out.print(value + ", ");	25
while (i < data.length) {	26	26	i++;	26
sum += data[i];	27	27	}	27
i++;	28	28	System.out.println();	28
}	29	29	// Print out the stats:	29
}	30	30	}	30
}	31	31	}	31
}	32	32	}	32
}	33	33	}	33
}	34	34	}	34

Accept Left

Accept Right

Apply

Abort

Git: предупреждение добавления «мусора»

- Часто появляются временные и вторичные файлы в директории
- Файл `.gitignore` используется для игнорирования таких файлов (wildcard синтаксис)
- Пример:

`# игнорим логи запусков`

`*.log`

`build/`

`temp-*`

Git: полезные команды

- **git init** — инициализация репозитория
- **git diff** — просмотр изменений в рабочей копии
- **git log** – показывает лог commits начиная с HEAD
- **git remote** – показывает информацию об удаленных репозиториях, а также позволяет удалять и добавлять их
- **git pull request** — запрос на изменение
- **git reset** — откат изменений рабочей копии
- **gitk / git gui** – визуальная утилита для работы с репозиторием
- Шпаргалка: <https://training.github.com/downloads/ru/github-git-cheat-sheet/>

Git: Установка и настройка

- ОС Linux: стандартные репозитории
apt-get install git
zypper in git
- MacOS: через Homebrew
brew install git
- Windows: <https://git-scm.com/>

Git: Первоначальная настройка

- Настройка информации о пользователе для всех локальных репозиториев
 - Устанавливает имя, которое будет отображаться в поле автора у выполняемых вами коммитов

```
$ git config --global user.name "[имя]"
```
 - Устанавливает адрес электронной почты, который будет отображаться в информации о выполняемых вами коммитах

```
$ git config --global user.email "[адрес электронной почты]"
```

Облачный репозиторий

- Специальный сервер — удаленный репозиторий кода
 - Предоставление интерфейса Git репозитория
 - Управление правами и ролями
 - Web интерфейс доступа в т.ч. к содержимому репозитория
 - Вики система
 - Система отслеживания проблем / Bug tracking system (Issues)
 - Continuous Integration/ Continuous Delivery (CI/CD)
 - Интеграция с экосистемой разработки
- Из самых известных это GitHub, GitLab, GitFlic, Bitbucket

Gitlab: окно входа

← → ↻ https://dev.cs.petrus.ru/users/sign_in 🔍 ☆ ⬇️ Ⓢ 📄 TOP ☰



GitLab Community Edition

LDAP Standard

Username

Password

Remember me

[Sign in](#)

[Explore](#) [Help](#) [About GitLab](#) [Community forum](#)

🌐 English ▾

Gitlab: проекты

← → ↻ <https://dev.cs.petsru.ru> ☆ ⬇ Ⓞ Ⓜ 🗨 ☰

  +  Your work / Проекты

📁 4 🧑‍🤝‍🧑 2 📧 5

🔍 Search or go to...

Your work

- 📁 Проекты
- 👤 Группы
- 🗨 Обсуждения 4
- 🧑‍🤝‍🧑 Запросы на слияние >
- 📧 Список задач 5
- 📁 Этапы
- ✂ Снимлеты
- 🕒 Активность

Проекты

Обзор проектов [Новый проект](#)

Yours 15 Starred 0 Personal Inactive

🕒 Search or filter results... 🔍 Обновлено ▾ ⚙

A	Maintenance / ansible-pull 🔒 Developer	☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 1 📁 0	Обновлено 1 неделю назад
C	Kirill Kulakov / Ctest 🔒 Владелец	✅ ☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 0 📁 0	Обновлено 3 недели назад
A	Maintenance / Ansible 🔒 Developer Ansible	☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 0 📁 0	Обновлено 1 месяц назад
G	Vadim Ponomarev / gitlab-test 🛡 Developer CI/CD sandbox	✅ ☆ 0 🧑‍🤝‍🧑 2 🧑‍🤝‍🧑 0 📁 0	Обновлено 2 мес. назад
O	Kirill Kulakov / Or Frequency Trend 🔒 Владелец	☆ 0 🧑‍🤝‍🧑 1 🧑‍🤝‍🧑 0 📁 0	Обновлено 1 год назад
C	Maintenance / cs-desktop 🔒 Developer	☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 0 📁 0	Обновлено 2 лет назад
A	Владислав Ермаков / adaptive-broker-system 🔒 Maintainer	☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 0 📁 0	Обновлено 2 лет назад
G	Maintenance / Guix Maintenance 🔒 Developer	☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 0 📁 0	Обновлено 2 лет назад
W	Kirill Kulakov / Web-SynDic 🔒 Владелец	☆ 0 🧑‍🤝‍🧑 0 🧑‍🤝‍🧑 0 📁 0	Обновлено 2 лет назад

 Помощь

Gitlab: просмотр проекта

The screenshot shows the GitLab interface for a project named 'Ctest'. The browser address bar shows the URL 'https://dev.cs.petsru.ru/kulakov/ctest'. The page header includes the GitLab logo, navigation icons, and the user profile 'Kirill Kulakov / Ctest'. Below the header, there are buttons for repository statistics (4 commits, 2 branches, 5 issues) and a search bar. The main content area features a sidebar with project navigation options like 'Pinned', 'Обсуждения', and 'Запросы на слияние'. The central pane displays the project name 'Ctest' with a lock icon, a commit history table, and a 'README.md' file view. The commit table lists files like 'app', 'tests', and '.gitignore' with their respective commit messages and dates. The 'README.md' section includes a 'Getting started' heading and introductory text. On the right, the 'Project information' sidebar shows details such as '12 Коммитов', '1 Ветка', and '728 КБ Project Storage'. At the bottom left, there is a 'Помощь' (Help) link.

← → ↻ 🔒 https://dev.cs.petsru.ru/kulakov/ctest ☆ ⬇️ Ⓞ 📄 🏠 🌐

+ 👤 Kirill Kulakov / Ctest

📄 4 🌐 2 📧 5

🔍 Search or go to...

Проект

- C Ctest
- 📌 Pinned
- Обсуждения 0
- Запросы на слияние 0
- 🔧 Manage
- 📅 Plan
- 📄 Code
- 🚀 Build
- 🛡️ Secure
- 🚚 Deploy
- 🏠 Operate
- 📺 Monitor
- 📊 Analyze
- ⚙️ Настройки

Ctest 🔒

🔔 В избранные 0 🍴 Fork 0 ⋮

🌐 main ctest / + 🔍 Найти файл 📄 Редактировать Код

Добавить LICENSE ✓ 4d71eb1a История
Kirill Kulakov создал 3 недели назад

Наименование	Последний коммит	Последнее обновление
📁 app	initial version	3 недели назад
📁 tests	fix test name	3 недели назад
🔥 .gitignore	initial version	3 недели назад
🔥 .gitlab-ci.yml	--ignore-errors mismatch	3 недели назад
📄 CMakeLists.txt	initial version	3 недели назад
🔥 LICENSE	Добавить LICENSE	3 недели назад
📄 README.md	Initial commit	3 недели назад

README.md

Ctest

Getting started

To make it easy for you to get started with GitLab, here's a list of recommended next steps.

Already a pro? Just edit this README.md and make it your own. Want to make it easy? [Use the template at the bottom!](#)

Project information

- 🔗 12 Коммитов
- 🌐 1 Ветка
- 🔗 0 Тегов
- 📄 728 КБ Project Storage
- 📄 README
- 🔗 MIT No Attribution
- 🔗 Конфигурация CI/CD
- + [Добавить CHANGELOG](#)
- + [Добавить CONTRIBUTING](#)
- + [Добавить кластер Kubernetes](#)
- + [Add Wiki](#)
- + [Configure Integrations](#)

Created on
January 10, 2025

🔗 Помощь

Gitlab: получение доступа к репозиторию

Найти файл Редактировать ▾ Код ▾

Клонировать с помощью SSH

`git@dev.cs.petrSU.ru:kulakov/cte` 

Clone with HTTP

`http://dev.cs.petrSU.ru/kulakov/` 

Open in your IDE

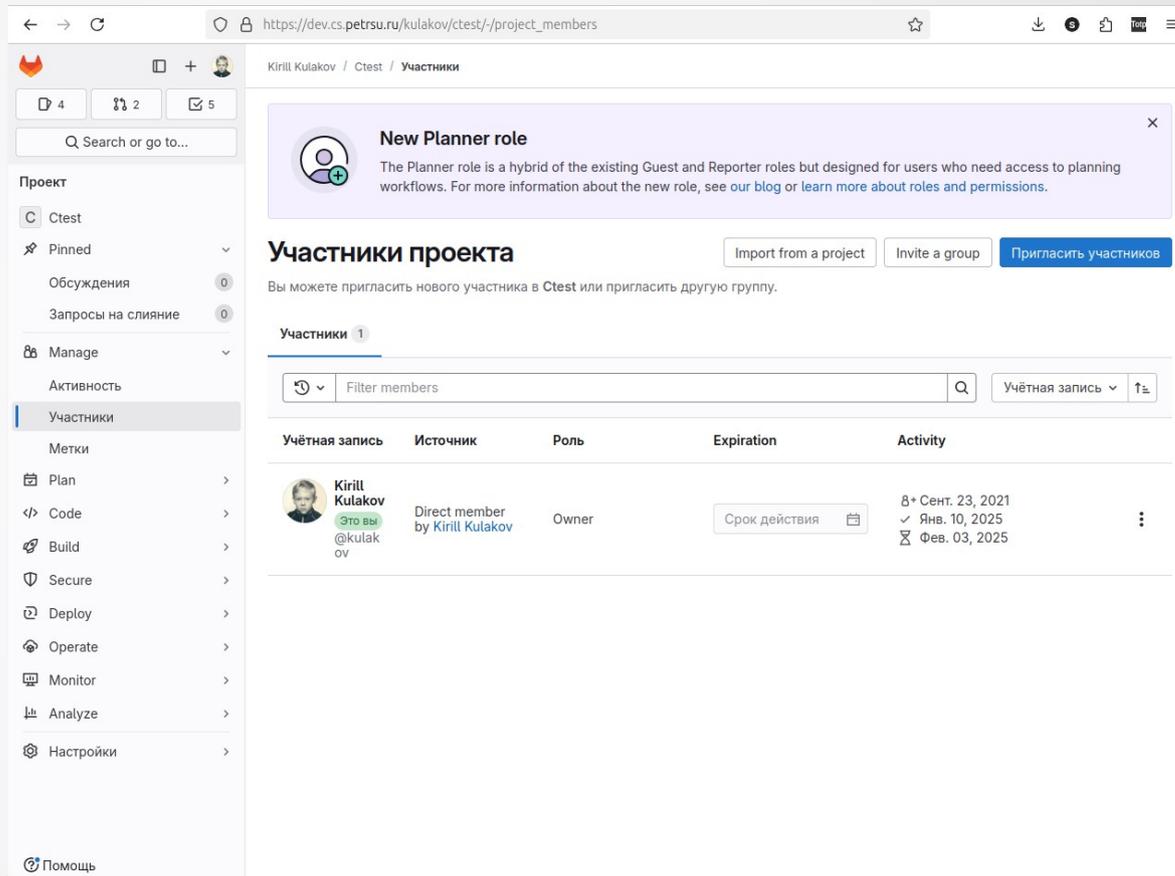
- Visual Studio Code (SSH)
- Visual Studio Code (HTTPS)
- IntelliJ IDEA (SSH)
- IntelliJ IDEA (HTTPS)

Скачать исходный код

- zip
- tar.gz
- tar.bz2
- tar

Gitlab: Пользователи и роли

- **Guest** - Гость
(доступ к приватным репозиториям)
- **Reporter** -
Репортер
- **Developer** -
Разработчик
- **Maintainer** -
Сопровождающий
- **Owner** - Владелец



The screenshot shows the GitLab interface for the 'Ctest' project. The left sidebar contains navigation options like 'Project', 'Manage', 'Plan', 'Code', 'Build', 'Secure', 'Deploy', 'Operate', 'Monitor', 'Analyze', and 'Settings'. The main content area displays a notification for a 'New Planner role' and a section for 'Участники проекта' (Project members). Below this, there is a table of project members.

Учётная запись	Источник	Роль	Expiration	Activity
 Kirill Kulakov Это вы @kulakov	Direct member by Kirill Kulakov	Owner	Срок действия	8+ Сент. 23, 2021 ✓ Янв. 10, 2025 ✗ Фев. 03, 2025

Доступ через ssh

The screenshot shows the GitLab user settings page for SSH keys. The browser address bar displays `https://dev.cs.petrus.ru/~user_settings/ssh_keys`. The page title is "Настройки пользователя / Ключи SSH".

Ключи SSH

Ключи SSH позволяют установить безопасное соединение между вашим компьютером и GitLab. SSH fingerprints verify that the client is connecting to the correct host. Check the [current instance configuration](#).

Your SSH keys 2 Добавить новый ключ

Заголовок	Ключ	Usage type	Created	Last used	Expires	Действия
kulakov@kulakov-huabook	<code>ssh-rsa c1:7a:90:4f:20:15:57:0c:7e:a8:06:8e:62:8f:8b:c6</code>	Authentication & Signing	3 недели назад	Никогда	Никогда	Отозвать Удалить
kulakov@zeta	<code>ssh-rsa f1:3a:d2:90:19:88:af:61:a6:ab:fa:99:b9:03:ca:c5</code>	Authentication & Signing	2 лет назад	2 лет назад	Никогда	Отозвать Удалить

Kirill Kulakov
@kulakov

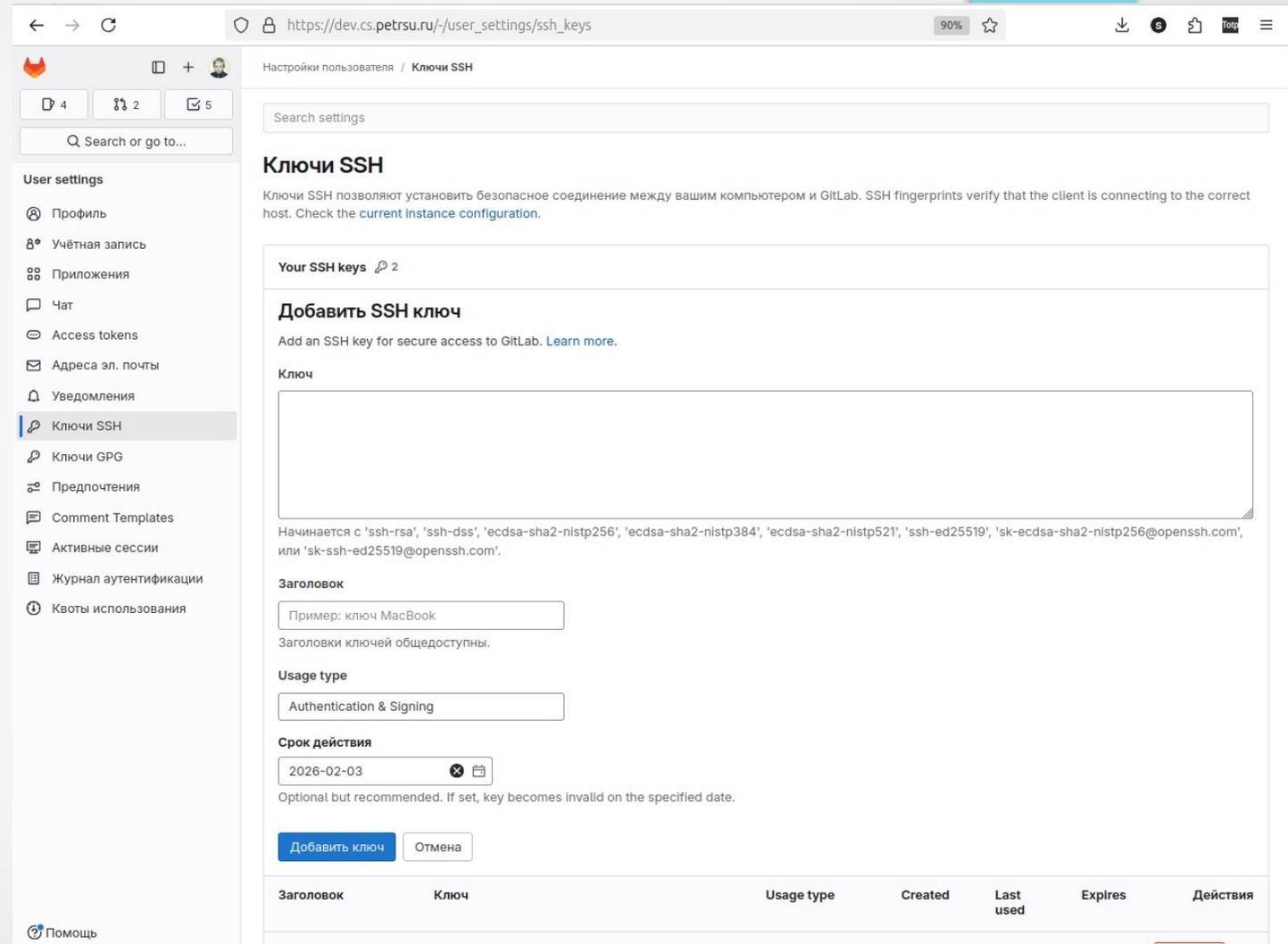
- Изменить статус
- Редактировать профиль
- Настройки
- Выйти

- User settings
- Профиль
 - Учётная запись
 - Приложения
 - Чат
 - Access tokens
 - Адреса эл. почты
 - Уведомления
 - Ключи SSH**
 - Ключи GPG
 - Предпочтения
 - Comment Templates
 - Активные сессии
 - Журнал аутентификации
 - Квоты использования

Доступ через ssh

- ssh-keygen

- ~/.ssh/
id_rsa.pub



The screenshot shows the GitLab user settings page for SSH keys. The browser address bar displays `https://dev.cs.petrso.ru/-/user_settings/ssh_keys`. The page title is "Настройки пользователя / Ключи SSH". A search bar is present at the top. The left sidebar contains a "User settings" menu with items like "Профиль", "Учётная запись", "Приложения", "Чат", "Access tokens", "Адреса эл. почты", "Уведомления", "Ключи SSH" (selected), "Ключи GPG", "Предпочтения", "Comment Templates", "Активные сессии", "Журнал аутентификации", and "Квоты использования".

The main content area is titled "Ключи SSH" and includes a sub-header "Your SSH keys 2". Below this is a "Добавить SSH ключ" section with the instruction "Add an SSH key for secure access to GitLab. Learn more." and a "Ключ" text area. A note specifies that the key must start with certain prefixes: "Начинается с 'ssh-rsa', 'ssh-dss', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', или 'sk-ssh-ed25519@openssh.com'".

Form fields include:

- Заголовок**: Input field with placeholder "Пример: ключ MacBook".
- Usage type**: Dropdown menu with "Authentication & Signing" selected.
- Срок действия**: Date picker set to "2026-02-03".

Buttons "Добавить ключ" and "Отмена" are at the bottom of the form. Below the form is a table with columns: "Заголовок", "Ключ", "Usage type", "Created", "Last used", "Expires", and "Действия".

Создание проекта

The screenshot shows a web browser window with the URL `https://dev.cs.petrus.ru/projects/new`. The page title is "Your work / Projects / New project". The main content area is titled "Создать новый проект" and contains three options:

- Создать пустой проект**: Create an empty project for storing files, planning work, and collaborative code writing.
- Создать из шаблона**: Create a project from a template with pre-filled files to start work quickly.
- Импортировать проект**: Import data from external repositories like GitHub, Bitbucket, or another GitLab instance.

At the bottom, there is a link: "Вы также можете создать проект из командной строки. [Показать команду](#)".

The left sidebar shows navigation options: "Проекты", "Группы", "Обсуждения" (4), "Запросы на слияние", "Список задач" (5), "Этапы", "Сниппеты", and "Активность". A search bar is also present.

At the bottom left, there is a "Помощь" (Help) link.

Создание проекта

← → ↻ https://dev.cs.petsru.ru/projects/new#blank_project 90% ☆ ⬇ Ⓢ 📄 🗨 ☰

  +  Your work / Projects / New project / Создать пустой проект

 4  2  5

🔍 Search or go to...

Your work

- 📁 Проекты**
- 👤 Группы
- 🗨 Обсуждения 4
- 🔗 Запросы на слияние >
- 📄 Список задач 5
- 📁 Этапы
- ✂ Сниметы
- 🕒 Активность

Создать пустой проект

Создайте пустой проект для хранения файлов, планирования работы и совместного написания кода.

Имя проекта

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Адрес Проекта **URL проекта**

Уровень доступа ⓘ

- 🔒 Приватный**
Project access must be granted to a group. Project is part of a group, access is granted to members of the group.
- 🔒 Внутренний**
The project can be accessed by internal users.
- 🌐 Публичный**
The project can be accessed by anyone.

Группы

- maintenance

Пользователи

- kulakov

Project Configuration

- Добавить README файл**
Позволяет вам сразу клонировать репозиторий. Пропустите этот пункт, если вы планируете загрузить существующий репозиторий.
- Включить Static Application Security Testing (SAST)**
Проанализируйте исходный код на наличие известных уязвимостей. [Learn more.](#)

 [Помощь](#)

Создание проекта

← → ↻ <https://dev.cs.petrso.ru/kulakov/test1#> 90% ☆

Kirill Kulakov / Test1

🔍 4 🧑 2 📧 5

🔍 Search or go to...

Проект

- Test1
- Pinned
- Обсуждения 0
- Запросы на слияние 0
- Manage
- Plan
- Code
- Build
- Secure
- Deploy
- Operate
- Monitor
- Analyze
- Настройки

Проект 'Test1' успешно создан.

Push files to this repository using SSH or HTTPS. If you're unsure, we recommend SSH.

SSH HTTPS

Создать новый репозиторий

```
git clone http://dev.cs.petrso.ru/kulakov/test1.git
cd test1
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main
```

Отправить существующую папку

Go to your folder

```
cd existing_folder
```

Configure the Git repository

```
git init --initial-branch=main
git remote add origin http://dev.cs.petrso.ru/kulakov/test1.git
git add .
git commit -m "Initial commit"
git push --set-upstream origin main
```

Отправить существующий репозиторий Git

Go to your repository

```
cd existing_repo
```

Configure the Git repository

```
git remote rename origin old-origin
git remote add origin http://dev.cs.petrso.ru/kulakov/test1.git
git push --set-upstream origin --all
git push --set-upstream origin --tags
```

Добавить участников в проект и начать сотрудничество с командой.

[Пригласить участников](#)

Upload File

- + Новый файл
- + Добавить README
- + Добавить LICENSE
- + Добавить CHANGELOG
- + Добавить CONTRIBUTING
- + Настройка CI/CD
- + Add Wiki
- + Configure Integrations

Created on
February 03, 2025

Помощь

Git Workflow: Создание обсуждения

← → ↻ <https://dev.cs.petsu.ru/kulakov/test1/-/issues/new> 90% ☆

Kirill Kulakov / Test1 / Обсуждения / New Issue

New Issue

Title (required)

Тип ⓘ

Обсуждение ▾

Описание

Предварительный просмотр | **B** *I* U | **≡** `</>` 🔗 📄 📑 🔍 🗑️ 📎 📧 📌

Write a description or drag your files here...

Switch to rich text editing [\[RI\]](#)

Add [description templates](#) to help your contributors to communicate effectively!

This issue is confidential and should only be visible to team members with at least the Planner role.

Assignee **Дата завершения**

Unassigned ▾ [Assign to me](#)

Этап

Выбрать этап ▾

Метки

Select label ▾

[Создать обсуждение](#) [Отмена](#)

Помощь

Git Workflow: запрос на слияние

The screenshot shows a web browser window displaying a merge request form. The browser's address bar shows the URL: `https://dev.cs.petrus.ru/kulakov/ctest/-/merge_requests/new?merge_request[issue_iid]=1&merge_request[...]`. The page title is "Новый запрос на слияние" (New Merge Request). The form is for a merge request from branch `1-add-coveralls-report` into branch `main`. The title field contains "Draft: Resolve 'add coveralls report'". The "Mark as draft" checkbox is checked, with a note: "Drafts cannot be merged until marked ready." The description field contains "Предварительный просмотр" (Preview) and "Closes #1". The assignee is set to "Kirill Kulakov", the reviewer is "Unassigned", and the stage is "Выбрать этап" (Select stage). The label is "Select label". The "Merge can start" dropdown is set to "Anytime". A footer link "Помощь" (Help) is visible in the bottom left corner.

Kirill Kulakov / Ctest / Запросы на слияние / Новый

Новый запрос на слияние

From `1-add-coveralls-report` into `main` [Сменить ветки](#)

Title (required)

Draft: Resolve "add coveralls report"

Mark as draft
Drafts cannot be merged until marked ready.

Описание

Предварительный просмотр | **B** *I* U | `</>` [🔗](#) [📌](#) [📋](#) [📄](#) [📁](#) [📎](#) [🗑️](#) [🔍](#)

Closes #1

Switch to rich text editing

Add [description templates](#) to help your contributors to communicate effectively!

Assignee

Kirill Kulakov

Reviewer

Unassigned

Этап

Выбрать этап

Метки

Select label

Merge can start

Anytime

Requires that merge checks pass.

Помощь

Git Workflow: после КОММИТОВ в ветку + CI/CD

← → ↻ https://gitlab.dckarelia.ru/opti-repair/web-api/-/merge_requests/354 ☆

Opti-Repair / Web API / Запросы на слияние / 1354

Resolve "Убрать дашборд, добавить датчики"

Слито Кулаков Кирилл requested to merge 18-ubrat-dashboard-dobavit... into dev 1 год назад

Редактировать Код

Обзор 0 Коммиты 2 Сборочные линии 3 Изменения 12 Add a to do

Closes #18 (closed)

👍 0 🗨️ 0 😊

✅ Сборочная линия #6957 пройдено

Сборочная линия пройдено для 6adac2c9 on 18-ubrat-dashboard-dobavit-datchiki 1 год назад
Test coverage 49.00% (0.00%) from 1 job

Approval is optional

Merged by Кулаков Кирилл 1 год назад Откатить Подобрать

Merge details

- Changes merged into dev with 4833a177.
- Deleted the source branch.
- Закрыт #18 (closed)
- Auto-merge enabled

✅ Сборочная линия #6958 пройдено

Сборочная линия пройдено для 4833a177 on dev 1 год назад
Test coverage 49.00% (0.00%) from 1 job

Активность

All activity ↑

- Кулаков Кирилл assigned to @kulakov 1 год назад
- Кулаков Кирилл added 1 commit 1 год назад

Ответственный Редактировать
Кулаков Кирилл

0 Reviewers Редактировать
Пусто - назначить себя

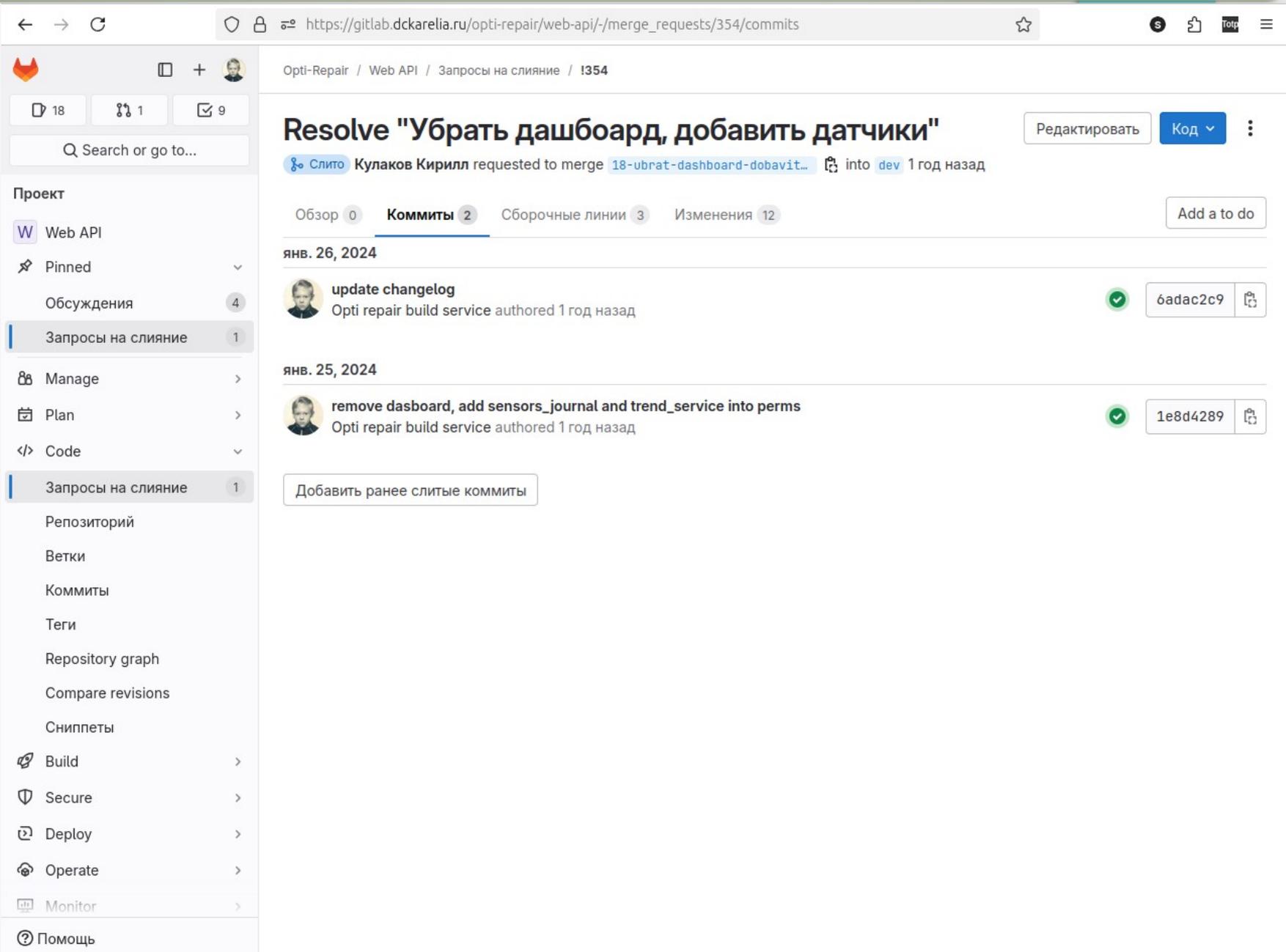
Метки Редактировать
Пусто

Этап Редактировать
Пусто

Учёт времени ⌚ +
Нет оценочного или потраченного времени

1 Participant

Git Workflow: просмотр коммитов запроса



The screenshot shows a GitLab merge request page for the repository 'Opti-Repair / Web API'. The merge request is titled 'Resolve "Убрать дашборд, добавить датчики"' and was requested by Кирилл Кулаков to merge branch '18-ubrat-dashboard-dobavit...' into the 'dev' branch. The page displays a list of commits associated with this merge request, including 'update changelog' and 'remove dasboard, add sensors_journal and trend_service into perms', both authored by 'Opti repair build service'.

← → ↻ https://gitlab.dckarelia.ru/opti-repair/web-api/-/merge_requests/354/commits ☆

Opti-Repair / Web API / Запросы на слияние / 1354

Resolve "Убрать дашборд, добавить датчики"

Редактировать Код ⌵ ⋮

Слито Кулаков Кирилл requested to merge `18-ubrat-dashboard-dobavit...` into `dev` 1 год назад

Обзор 0 **Коммиты 2** Сборочные линии 3 Изменения 12 Add a to do

янв. 26, 2024

 **update changelog**
Opti repair build service authored 1 год назад ✓ `6adac2c9` 

янв. 25, 2024

 **remove dasboard, add sensors_journal and trend_service into perms**
Opti repair build service authored 1 год назад ✓ `1e8d4289` 

Добавить ранее слитые коммиты

Проект

- Web API
- Pinned
- Обсуждения 4
- Запросы на слияние 1**
- Manage
- Plan
- Code
- Запросы на слияние 1**
 - Репозиторий
 - Ветки
 - Коммиты
 - Теги
 - Repository graph
 - Compare revisions
 - Сниппеты
- Build
- Secure
- Deploy
- Operate
- Monitor
- Помощь

Git Workflow: просмотр изменений запроса

← → ↻ https://gitlab.dckarelia.ru/opti-repair/web-api/-/merge_requests/354/diffs ☆

Opti-Repair / Web API / Запросы на слияние / 1354

Resolve "Убрать дашборд, добавить датчики"

Редактировать Код ⋮

Слито Кулаков Кирилл requested to merge 18-ubrat-dashboard-dobavit... into dev 1 год назад

Обзор 0 Коммиты 2 Сборочные линии 3 **Изменения 12** Add a to do

Compare dev and последняя версия 12 файлов +63 -52

app/enums/security.py +8 -6

```
@@ -61,17 +61,18 @@ class ScopePermission(IntEnum):
61 61
62 62
63 63     user_scopes_descriptions = {
64 -     'DASHBOARD': "Дашборд",
65 -     'COOLANT_SERVICE': "Сервис контроля СОЖ",
66 -     'WORKLOAD_SERVICE': "Сервис загрузки оборудования",
67 64     'NODES_JOURNAL': "Журнал узлов",
68 65     'MODULES_JOURNAL': "Журнал программных модулей",
66 +     'SENSORS_JOURNAL': "Журнал датчиков",
69 67     'MEASUREMENTS_JOURNAL': "Журнал измерений",
70 68     'USERS_JOURNAL': "Журнал пользователей",
71 69     'ROLES_JOURNAL': "Журнал ролей",
72 70     'DEVELOPER_MODULE_INFO': "Зона разработчика программного
73 71     модуля",
74 72     'HARDWARE_JOURNAL': "Журнал аппаратных модулей",
73 +     'TREND_SERVICE': "Сервис построения трендов",
74 +     'COOLANT_SERVICE': "Сервис контроля СОЖ",
75 +     'WORKLOAD_SERVICE': "Сервис загрузки оборудования",
76 76     'DIAGNOSTIC_ASSISTANCE_SERVICE': "Сервис помощи в диагностике",
77 77     'VIBRATION_DIAGNOSTICS_SERVICE': "Сервис вибродиагностики",
78 78     }
@@ -81,17 +82,18 @@ class UserScopesEnum(Enum):
81 82     """
82 83     Класс-перечисление прав пользователя.
83 84     """
84 -     DASHBOARD = auto() # право работы с дашбордом
85 -     COOLANT_SERVICE = auto() # право на работу с загрузенностью
86 -     WORKLOAD_SERVICE = auto() # право на работу с загрузенностью
```

Файлы 12

Search (e.g. *.vue) (Ctrl+P)

- app
 - enums
 - security.py** +8 -6
 - messaging
 - broker_event_bus.py +4 -1
 - routers
 - dashboard
 - widgets.py +1 -1
 - modules
 - dtcp
 - dtcp_a... dules.py +4 -4
 - dtcp_e... dules.py +4 -4
 - dtcp_f... dules.py +4 -4
 - dtcp_t... dules.py +4 -4
 - commons.py +4 -1
 - module... ements.py +2 -1
 - nodes_rel.py +4 -1
 - auth.py +17 -25