

Engineering Open Source Projects

Anton Grishin ([@alchemmunist](https://twitter.com/alchemmunist))

Введение в курс EOSP.



Содержание

`hello` Лучше познакомимся друг с другом.

`open-source` Погружаемся в мир OpenSource.

`course` Разберём, что нас ждёт на курсе.

`practice` Придумаем название для проекта вместе.

`live-demo` Закладываем основу проекта вместе.

`api` Познакомимся с GitHub API.

`github-flow` Как организованы процессы разработки Open Source.

Давайте познакомимся

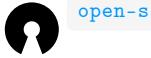
hello

Что здесь неверно?

- Я Антон. Учусь на 2 курсе SWE в Central University.
- Участвовал в разработке 70+ репозиториев и сделал ~1600 коммитов.
- Я выпускник Яндекс Лицей с золотым сертификатом.
- Использую Linux как основную операционную систему последние 3 года.
- ~~Я могу печатать 120 слов в минуту на клавиатуре qwerty.~~
- Раньше был профессиональным волейболистом.
- Использую только терминал для разработки.
- Автор блога: alchemmist.xyz

Расскажите о себе 

Что такое Open Source



Исходный код проекта открыт для всех.

Можно копировать? Можно использовать? Можно продавать?

Postgres DBMS (MIT/BSD)

open-source

- Полный доступ к исходному коду
- Использование в коммерческих продуктах
- Продажа как часть своего продукта
- Закрытие своего кода поверх PostgreSQL
- Сохраняйте лицензию и имя автора

postgres

Overview Repositories 5 Projects Packages People 6

 **PostgreSQL**
Verified
1.8k followers Worldwide https://www.postgresql.org/

Pinned

 **postgres** Public
Mirror of the official PostgreSQL GIT repository. Note that this is just a *mirror* - we don't work with pull requests on github. To contribute, please see https://wiki.postgresql.org/wiki/Submitting_a_Patch...
● C ⭐ 19.6k ⚡ 5.3k

Repositories

Find a repository... Type Language

 **postgres** Public
Mirror of the official PostgreSQL GIT repository. Note that this is just a *mirror* - we don't work with pull requests on github. To contribute, please see https://wiki.postgresql.org/wiki/Submitting_a_Patch...
● C ⭐ 19.554 ⚡ 5.329 ○ 0 ⚡ 0 Updated 15 hours ago

 **pgweb** Public
Mirror of the code behind www.postgresql.org

HTML ⭐ 80 ⚡ 48 ○ 0 ⚡ 0 Updated 3 days ago

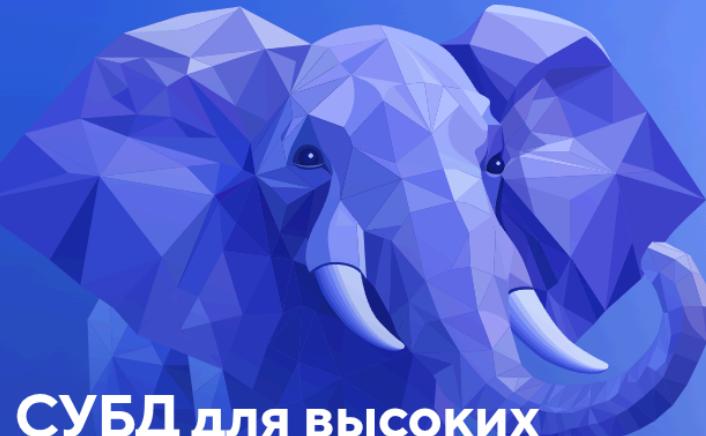
 **pgcommitfest** Public
The source code for <https://commitfest.postgresql.org>

Python ⭐ 12 ⚡ 19 ○ 13 ⚡ 1 Updated 3 days ago

Postgres PRO (EULA)

open-source

- EULA — лицензионное соглашение с конечным пользователем
- Распространяется за деньги
- Модель OpenCore



**СУБД для высоких
нагрузок и эффективной
работы бизнеса**

Весь опыт
разработчиков
PostgreSQL для вас

Angular JS (MIT)

[open-source](#)

- Полностью открыт (как Postgres)
- Репозиторий существует, но устарел
- Нет поддержки
- Нет улучшений
- Нет исправлений ошибок
- Причина — переход на TypeScript

This repository was archived by the owner on Apr 12, 2024. It is now read-only.

 angular.js [Public archive](#)

[Watch 3738](#) [Fork 27.3k](#) [Star 59k](#)

[master](#) [Go to file](#) [Code](#) [About](#)

 Brocco	chore: update post LT...	d8f7781 · 2 years ago
 .circleci	chore(build): supp...	5 years ago
 .github	chore(*): prep for ...	6 years ago
 benchmarks	chore(benchpress...	8 years ago
 css	style(css) separat...	9 years ago
 docs	chore: update pos...	2 years ago
 i18n	chore(i18n): fix U...	8 years ago
 images	docs(*): optimize i...	9 years ago
 lib	chore(ci): deploy t...	5 years ago
 logs	creating logs/ and...	16 years ago
 scripts	chore(functions): ...	5 years ago
 src	docs(\$parse): fix t...	5 years ago
 test	test(Angular): fix a...	5 years ago

AngularJS - HTML enhanced web apps!

[angularjs.org](#)

[Readme](#) [MIT license](#) [Code of conduct](#) [Contributing](#) [Security policy](#) [Activity](#) [Custom properties](#)

 59k stars  3.7k watching  27.3k forks [Report repository](#)

Releases

 209 tags

Ядро Linux (GPLv2)

open-source

- Полный доступ к исходному коду
- Использование в коммерческих продуктах
- Форки должны оставаться под GPL при распространении: копильтф

torvalds / linux

Code Pull requests Actions Projects Security Insights

linux Public

master 1 Branch 914 Tags Go to file Add file

torvalds Merge tag 'erofs-for-6.19-rc5-fixes' of git://git.kernel.org/pub/scm/... b6151c4 · 7 hours ago

Documentation Merge tag 'soc-fixes-6.19' of git://git.kernel.org/pub/scm...

LICENSES LICENSES: Add modern form of the LGPL-2.1 tags to the ...

arch Merge tag 'arm64-fixes' of git://git.kernel.org/pub/scm/li...

block Merge tag 'block-6.19-20260109' of git://git.kernel.org/p...

certs sign-file,extract-cert: use pkcs11 provider for OPENSSL M...

crypto crypto: seqiv - Do not use req->iv after crypto_aead_encr...

drivers Merge tag 'block-6.19-20260109' of git://git.kernel.org/p...

fs Merge tag 'erofs-for-6.19-rc5-fixes' of git://git.kernel.org/...

include Merge tag 'acpi-6.19-rc5' of git://git.kernel.org/pub/scm/...

init Merge tag 'mm-nonmm-stable-2025-12-06-11-14' of git:/...

io_uring Merge tag 'io_uring-6.19-20260109' of git://git.kernel.org...

ipc Merge tag 'mm-nonmm-stable-2025-12-06-11-14' of git:/...

kernel Merge tag 'pm-6.19-rc5' of git://git.kernel.org/pub/scm/li...

lib idr: fix idr_alloc() returning an ID out of range

mm mm/ksm: fix pte_unmap_unlock of wrong address in bre...

net Merge tag 'ceph-for-6.19-rc5' of https://github.com/ceph...

OpenSource — основа современной IT-индустрии

open-source

Пространство, где появляются технологии.



«99% компаний из Fortune 500 используют открытое ПО. <...> Более 56 миллионов разработчиков вносят вклад в проекты Open Source. <...> Из-за растущей нагрузки рынок операционных систем Linux ожидается рост на 7% в год, достигнув \$9,7 млрд к 2024 году.»

Pranay Ahlawat, Boston Consulting Group

Article «Why You Need an Open Source Software Strategy», Апрель 2021

Идея и цель нашего проекта

course

Курс полностью практико-ориентированный.

- Построить практическую систему оценки вклада разработчиков на основе активности в GitHub
- Изучить модульный дизайн ПО: library → CLI → Telegram bot
- Практиковать реальные Open Source workflow: issues, pull requests, reviews и т.д.
- Сосредоточиться на чистом, поддерживаемом и тестируемом коде
- Испытать CI/CD пайплайны, релизы и автоматизацию деплоя
- Документировать, настраивать и организовывать проекты правильно
- Развить навыки создания презентаций и публичной демонстрации проекта

Два основных сценария

course

Profile analytics

- HR хочет быстро получить информацию о деятельности разработчика без ручного просмотра GitHub
- Анализировать **весь профиль GitHub**: все репозитории, вклад и история активности
- Понять, какие языки и технологии использует разработчик
- Отслеживать вклад по репозиториям: коммиты, pull requests, issues
- Составить краткое резюме профиля для рекрутинговых решений

Leader board of team

- Руководители команд хотят видеть продуктивность команды
- Анализировать **вклад внутри одного репозитория** для справедливого сравнения членов команды
- Отслеживать метрики каждого разработчика: качество кода, участие в реview, решение задач
- Выявлять, кто активно вносит вклад, а кто нуждается в поддержке
- Предоставлять честные, основанные на данных рекомендации для улучшения взаимодействия в команде

Три настройки, три проекта

course

Python library

- Вычислять метрики вклада разработчика из данных GitHub
- Предоставлять переиспользуемые, модульные функции для расчёта метрик
- Включать комплексные unit-тесты и следовать TDD
- Служить основой для CLI и интеграции с ботом
- Поддерживать лёгкое расширение и сопровождение

CLI

- Предоставлять командный доступ к библиотеке метрик
- Поддерживать несколько команд, флагов и опций
- Удобно получать, отображать и экспорттировать данные
- Корректно обрабатывать ошибки и показывать информативные сообщения
- Интеграция с CI/CD для автоматических релизов

Telegram bot

- Обеспечивать удобный доступ к метрикам через интерфейс Telegram
- Взаимодействовать с пользователями, обрабатывать команды и запросы
- Безопасно управлять секретами и токенами API
- Получать данные из библиотеки и форматировать их для удобного отображения
- Поддерживать уведомления, обновления и автоматические оповещения

Как воспринимать этот курс?

course

Мотивация и настрой.



«Грустно думать о том, что все школьники отворачиваются от строительства домиков на деревьях и сидят в классе, усердно изучая Дарвина или Ньютона ради экзамена, в то время как работа, сделавшая Дарвина и Ньютона знаменитыми, по духу ближе к строительству домиков на деревьях, чем к учебе ради экзамена.»

Paul Graham

Essay «A Project of One's Own», Июнь 2021

Пора придумать название!

practice

Перейдите к Figma board!

Сделаем первый шаг

[live-demo](#)

Создание GitHub организации и репозитория.

Введение в GitHub API.

api

```
gh api /octocat
```

GitHub API — это просто HTTP api

Любой инструмент, умеющий отправлять HTTP-запросы, может работать с GitHub API.

- `gh` — удобная оболочка для API
- `curl` — raw HTTP из терминала
- `Python` — программный доступ для автоматизации и логики

Официальный gh cli api

Утилита CLI для работы со всеми возможностями GitHub из терминала.

Установка в shell:

```
# Mac:  
brew install gh  
  
# Windows:  
winget install --id GitHub.cli  
  
# Arch:  
sudo pacman -S github-cli
```

И попробуйте что-то, например:

```
gh api /users/alchemmist
```

В результате:

```
1  {  
2      "login": "alchemmist",  
3      "avatar_url": "https://avatars.githubusercontent.com/u/104511335?v=4",  
4      "html_url": "https://github.com/alchemmist",  
5      "followers_url": "https://api.github.com/users/alchemmist/followers",  
6      "subscriptions_url": "https://api.github.com/users/alchemmist/subscriptions",  
7      "repos_url": "https://api.github.com/users/alchemmist/repos",  
8      "type": "User",  
9      "name": "Anton Grishin",  
10     "blog": "alchemmist.xyz?utm_source=github",  
11     "location": "Russia, Moscow",  
12     "email": "anton.ingrish@gmail.com",  
13     "followers": 18,  
14     "created_at": "2022-04-27T14:12:26Z",  
15     "updated_at": "2026-01-09T06:23:55Z",  
16     ...  
17 }
```

Подробнее в документации

api

Используем curl для GitHub API

api

Raw HTTP-запросы из терминала.

Отправка GET с аутентификацией:

```
curl -L \
-H "Accept: application/vnd.github+json" \
-H "Authorization: Bearer <TOKEN>" \
-H "X-GitHub-Api-Version: 2022-11-28" \
https://api.github.com/repos/alchemmist/eosp/stats/contributors
```

Согласно документации:

- `w` — начало недели, задано как Unix timestamp
- `a` — количество добавлений
- `d` — количество удалений
- `c` — количество коммитов

Вс, 14 Дек 2025

В результате:

```
1  [
2    {
3      "total": 37,
4      "weeks": [
5        {
6          "w": 1765670400,
7          "a": 6477,
8          "d": 0,
9          "c": 1},
10         {"w": 1768089600,
11          "a": 0,
12          "d": 0,
13          "c": 0}, ...
14     ],
15     "author": {
16       "login": "alchemmist",
17       "id": 104511335,
18       "node_id": "U_kgDOBjq3Zw", ...
19     }
20   ]
```

Подробнее в документации

api

GitHub API с Python

api

Эффективные, параллельные, продакшин-запросы.

Пример с `httpx`:

```
import asyncio, httpx

async def fetch_prs(username):
    url = f"https://api.github.com/search/issues?" \
          f"q=author:{username}+type:pr+created:>2025-01-01"
    async with httpx.AsyncClient() as client:
        resp = await client.get(
            url,
            headers={
                "Authorization": "Bearer <TOKEN>"
            },
        )
        data = resp.json()
        for pr in data["items"]:
            print(f"{pr['title']}\n\t-> {pr['html_url']}")

asyncio.run(fetch_prs("alchemmist"))
```

*GraphQL - Результат:

язык

запросов

для API,

позволяющий

запрашивать

ровно

необходимые

данные в

одном

запросе,

без лишних

полей.

Add alchemmist.xyz individual blog

-> https://github.com/kilimchoi/engineering-blogs/pull/1201

Add alchemmist.xyz personal blog

-> https://github.com/learnAnything/blogs/pull/21

Add alchemmist.xyz blog

-> https://github.com/logancyang/awesome-personal-websites/pull/1

Add alchemmist.xyz blog

-> https://github.com/jkup/awesome-personal-blogs/pull/173

Add @alchemmist_blog to personal blogs section

-> https://github.com/goq/telegram-list/pull/992

Add @alchemmist_blog to personal blogs section

-> https://github.com/alchemmist/telegram-list/pull/1

Add a "quiet" exit (#104)

-> https://github.com/cqfn/aibolit/pull/818

Ограничения GitHub API

api

- Rate limit: **5000 запросов/час** для аутентифицированных
- Rate limit: **60 запросов/час** для неаутентифицированных
- Пагинация: максимум **100 элементов на страницу**, нужно обрабатывать страницы
- Приватные данные требуют корректной аутентификации и прав (scopes)
- GraphQL vs REST: иногда проще через GraphQL, но сложные запросы могут достигать лимитов
- Ответы API могут кешироваться; для актуальных метрик возможны повторные запросы
- Эндпоинты могут меняться; библиотека должна учитывать версионирование API

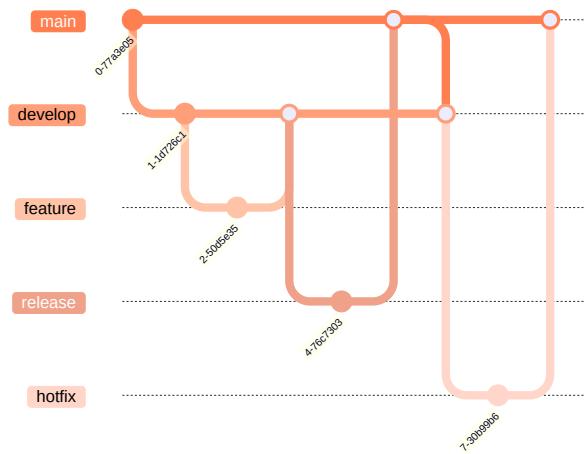
Workflow разработки

github-flow

Два подхода к разработке.

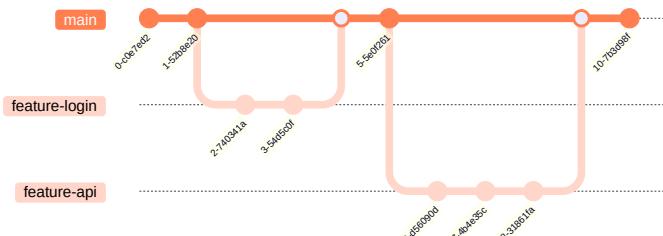
Git Flow

Структурированная, процессно-тяжёлая модель ветвления, рассчитанная на запланированные релизы. Долгоживущие ветки, явное управление релизами → предсказуемо, но медленно адаптируется.



GitHub Flow

Лёгкий workflow для continuous delivery и Open Source. Ветка `main` всегда готова к деплою; все изменения проходят через pull requests.



Почему GitHub Flow? github-flow

- Все изменения через **pull requests** → code review и CI/CD проверки
- Поощряет маленькие инкрементальные изменения, вместо долгих веток
- Ветка `main` всегда готова к деплою
- Интеграция с issues и project boards → планирование и трекинг
- Прозрачность: команда может комментировать, ревьюить, одобрять или отклонять изменения

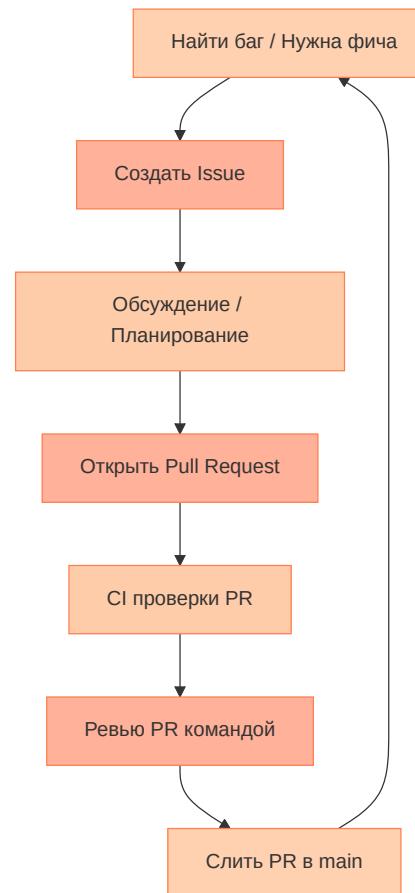
Основные сущности GitHub Flow github-flow

Issue	Branch	Commit
Описывает баг, фичу, задачу или вопрос. Начальная точка для разработки.	Изолированная рабочая ветка для конкретной фичи или исправления.	Отдельные изменения, отслеживаемые в истории Git.
Pull Request (PR)	Code Review	CI/CD Checks
Предлагает изменения из ветки в main. Облегчает ревью и обсуждение.	Команда проверяет PR, чтобы обеспечить качество и поддерживаемость кода.	Автоматические тесты, линтеры, сборка и пайплайны деплоя.
Merge		Одобренный PR сливаются в main и обычно запускает деплой.

GitHub Flow на практике

github-flow

1. Найти баг или фичу → **создать issue**
2. Создать ветку от `main` для этой issue
3. Делать **коммиты** и пушить на GitHub
4. Открыть **pull request**, связав с issue
5. Code review и автоматические CI/CD проверки
6. После одобрения PR сливаются в `main`
7. Деплой запускается автоматически (если настроено)



Лучшие практики GitHub Flow

github-flow

- Короткоживущие ветки → частая интеграция снижает конфликты
- Понятные коммиты → информативная история
- Ссылки на issues в PR → контекст
- Шаблоны PR и issues → стандартизация
- Автоматизация → CI/CD, тесты, линтеры, проверки
- Культура реview → лучший код, обмен знаниями, ответственность

