то такое GitHub и чем он отличается от Git

В 2005 году создатель ядра Linux Линус Торвальдс столкнулся с проблемой: с ним над проектом работало много людей, каждый делал свою часть — и надо было управлять добавлением нового кода, тестированием и устранением багов.

Так знаменитый разработчик придумал Git — распределённую систему управления версиями. Её устанавливают на все компьютеры, где ведётся работа над проектом. Система позволяет создавать и развивать ветви проекта, откатываться к предыдущим версиям, клонировать. Подробнее про Git мы писали в [отдельной статье](https://academy.yandex.ru/journal/chto-takoe-git?utm_source=academy_site&utm_medium=article&utm_campaign=journal&utm_content=03.07.2023).

[GitHub](https://github.com/) — это облачный хостинг репозиториев Git, или папок, где Git отслеживает изменения. Это Google Фото, только не для картинок, а для репозиториев. Но GitHub не только хранилище файлов с кодом. Эта платформа также предлагает безопасную авторизацию по токенам, инструменты тестирования и анализа кода, сервисы деплоя проектов — GitHub Actions.

В GitHub можно подсматривать за чужим кодом в опенсорсных проектах, комментировать, копировать то, что понравилось, предлагать свои решения. Здесь создают личные портфолио проектов. GitHub называют социальной сетью для разработчиков.

Платформа бесплатная, но только для опенсорсных проектов. Если захотите разместить приватный проект, к которому будет доступ только у вас и вашей команды, нужно купить подписку.

Разбираемся с терминами

**Репозиторий** — папка с вашим проектом, где лежат файлы, иконки программы, разные картинки. В GitHub у каждого репозитория есть страница с описанием проекта.

**Ветка** (Branch) — копия проекта в рамках одного репозитория. Есть главная ветка проекта, или main. Но любой разработчик может скопировать проект в свою ветку и работать над его частью, не трогая исходный код и не мешая другим разработчикам. Ветки независимы друг от друга, но их можно объединять, мёржить (от англ. merge — слияние), даже если есть разница в коде.

**Клонирование** — копирование репозитория из GitHub на жёсткий диск. При клонировании на компьютер пишется вся история версий, все ветки. Если кто-то вносит изменения в репозиторий, вы их тоже получите. Простое копирование таких возможностей не даёт.

**Коммит** — внесение изменений в репозиторий, чтобы их увидели другие разработчики. У каждого коммита есть временная метка и хеш-сумма.

**Форк** — копирование репозитория, обычно чужого, для продолжения разработки по другому пути. Часто бывает в опенсорсных проектах.

**Пул-реквест** (pull request) — предложение автору проекта своих улучшений, чтобы он залил их в исходный репозиторий.

Как работать на GitHub

**Через браузер** — так проще начать изучение GitHub, не надо учить команды для терминала. Так как GitHub — хостинг для репозиториев, первый шаг при погружении в платформу — создание своего репозитория.

Выберите пункт New Repository, заполните название и краткое описание, проставьте нужные галочки и щёлкните на Create Repository

На сайте можно выполнять все основные действия с репозиторием: клонировать, форкать, мёржить ветки, просматривать и разрешать конфликты, удалять проекты. Если нужно внести небольшие изменения в проект или вы пользуетесь GitHub редко, сайта достаточно.

**Работа с GUI-клиентом** [GitHub Desktop](https://desktop.github.com/" \t "_blank) — следующий уровень погружения в GitHub. Он не требует знания команд для терминала, но позволяет удобно настроить работу с платформой и полноценно заниматься написанием кода в локальной копии репозитория у себя на компьютере.

Клиент позволяет клонировать репозиторий из хостинга себе на компьютер и отправлять локальные репозитории обратно. Он похож на клиент Яндекс Диска: синхронизирует ваши файлы с облаком.

Но только [GitHub CLI](https://cli.github.com/" \t "_blank) **(Command Line Interface)** позволяет использовать все возможности платформы. В Linux (например, в Ubuntu) CLI Git установлен «из коробки». Иногда предустановлен CLI и в MacOS. В Windows для ввода команд можно использовать PowerShell.

Для работы вам пригодится [шпаргалка по консольным командам](https://github.com/cyberspacedk/Git-commands).

Как посмотреть чужие репозитории

Многие ваши задачи кто-то уже решил. Например, Яндекс написал асинхронный фреймворк [userver](https://userver.tech/" \t "_blank) для быстрого создания микросервисов, сервисов и утилит на C++. Почему бы не воспользоваться чужими наработками?

Вот что можно увидеть на главной странице репозитория — на примере фреймворка Яндекса userver

Справа — краткое описание проекта. Видно, что у репозитория 15 веток, в ветке master 3801 коммит, 18 пул-реквестов, последний релиз был вчера. У проекта лицензия Apache-2.0.

На что обращать внимание при поиске нужных решений в других проектах:

* Описание на главной странице под списком файлов.
* Лицензия проекта — [GPLv3](http://www.gnu.org/licenses/gpl.html) или [Apache 2.0](http://directory.fsf.org/wiki/License:Apache2.0). В ней указано, на каких условиях вы можете его использовать.
* Популярность и дата последнего обновления. В редких случаях может подойти инструмент, который написан десять лет назад, но чаще нужен проект, который поддерживается автором или сообществом разработчиков. На это указывает частота обновлений, количество и содержание коммитов, форков. Посмотрите также на звёздочки — это некий аналог лайков и репостов.

Как оформить свой профиль на GitHub

Разработчики используют аккаунт GitHub как портфолио. Чтобы он привлекал работодателей, важно правильно его оформить. Вот несколько советов:

1. Используйте в профиле свои реальные имя и фамилию и поставьте качественную фотографию. Но личную электронную почту лучше не публиковать, чтобы не получать много спама.
2. Размещайте в открытом доступе только те репозитории, что вы хотите показать потенциальному работодателю.
3. Описывайте каждый проект как можно более точно и подробно, используя файл [Readme.md](http://readme.md/).
4. Посмотрите во вкладке Trending, как оформлены профили самых трендовых разработчиков GitHub, и пользуйтесь лучшими практиками.