Тема 1.3 Git и GitHub

**Что такое Git и GitHub?**

Git — это система управления версиями, которая пришлась по душе практически всем — от разработчиков до дизайнеров. GitHub можно считать соцсетью для хранения кода. Это настоящая Мекка для технарей. Здесь вы можете попрактиковаться в разработке и придумать что-то свое, найти множество open-source проектов, передовых технологий, различных функций и дизайнов.

На GitHub вы учитесь и участвуете в других проектах, храните код для работы или учебы, берете код других проектов и вникаете во все детали. А еще вы можете создавать сайты бесплатно напрямую из репозитория!

GitHub неразрывно связан с системой контроля версий Git, которую разработчики устанавливают на персональный компьютер. На базе Git есть платформа GitHub, где хранятся git-репозитории с открытым кодом. Это позволяет командам работать совместно. Чтобы разобраться, как всё устроено в GitHub, нужно понимать, зачем нужна система контроля версий Git (сокращённо VCS от англ. Version Control System).

Что такое Git

Чтобы команда могла совместно работать над одним файлом, нужен сервис, где будет храниться папка с файлами — репозиторий.

Сервер видит, какие изменения вносит каждый член команды, и контролирует, нет ли конфликтующих между собой частей кода. Такие VCS называют централизованными. Это удобнее, чем постоянно делать резервные версии. Но если сервер будет неисправен, все наработки пропадут.

Поэтому удобнее воспользоваться такими сервисами, как Git, — с распределёнными VCS. Файлы синхронизируются между ПК и центральным репозиторием. Каждый программист получает весь репозиторий, а не конкретные файлы. Можно как отправить изменения на сервер, так и скачать те, что внесли другие программисты.

Ревьюер — обычно коллега-программист — одобряет запрос на добавление кода, и тогда этот код становится частью репозитория. Если есть замечания, ревьюер пишет комментарии, и разработчик их учитывает.

В случае с распределёнными VCS актуальные файлы можно взять у любого из членов команды. И даже если что-то вышло из строя у одного из разработчиков, это не страшно.

**В чём разница между Git и GitHub**

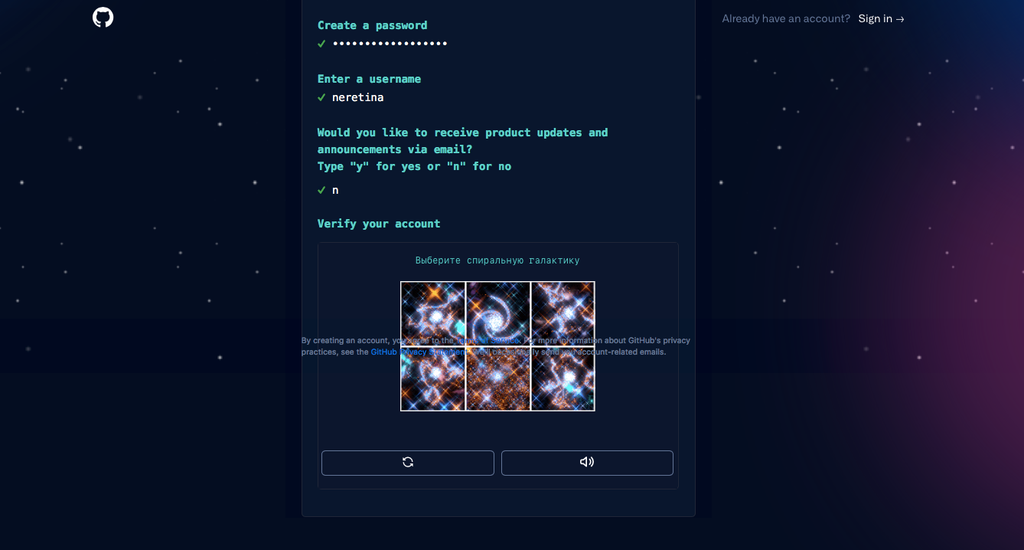
С помощью Git программисты и разработчики ориентируются в коде и отслеживают изменения. Git помогает вернуть файлы в исходное состояние и видеть изменения, внесённые в определённый период. Разработчик выполняет разные команды (например, commit, push), а все изменения синхронизируются с центральным репозиторием.

Git — это система контроля версий, а GitHub — онлайн-сервис, по сути социальная сеть. Одна из основных целей GitHub — быть единым местом для проектов с исходным кодом. Предполагается, что пользователь делится чем-то полезным, а другие люди смогут участвовать в разработке.

Ещё один вариант — использовать GitHub как хранилище проектов для портфолио: легко дать на них ссылку.

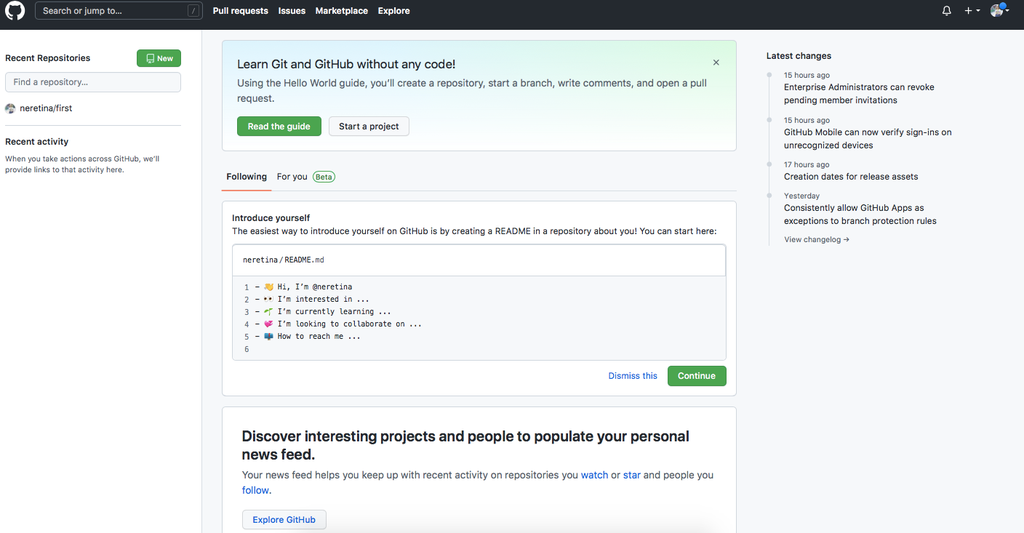
**Создание аккаунта в GitHub**

Чтобы выложить на GitHub проект, нужна регистрация на сайте сервиса. Первоначально необходимо указать адрес электронной почты, который станет логином для учётной записи. Сюда будут приходить все уведомления, и с помощью электронной почты можно будет восстановить доступ к GitHub в случае необходимости.



При регистрации попросят указать email, имя пользователя, согласиться на рассылку или отказаться

После регистрации открывается панель управления, где будет отображаться статистика и созданные репозитории.



Дашборд помогает увидеть последние изменения и статистику работы с проектами

**Создание репозитория**

Репозиторий используют для организации одного проекта. В нём содержатся файлы с кодом, необходимые для разработки проекта. Есть приватные и публичные репозитории. К приватным можно подключить ограниченное число человек, публичные легко находятся в поиске другими пользователями.

Репозитории могут включать файл README с информацией о проекте. Для загрузки на GitHub своего репозитория добавлять файл README не нужно. В этом случае нужен именно пустой репозиторий, куда будет залит существующий код. С пустым репозиторием на GitHub отобразит подсказку, как создать или клонировать репозиторий.

Чтобы загрузить или создать репозиторий, воспользуйтесь выпадающим меню в правом верхнем углу на любой из страниц GitHub и выберите New repository.

Создание репозитория

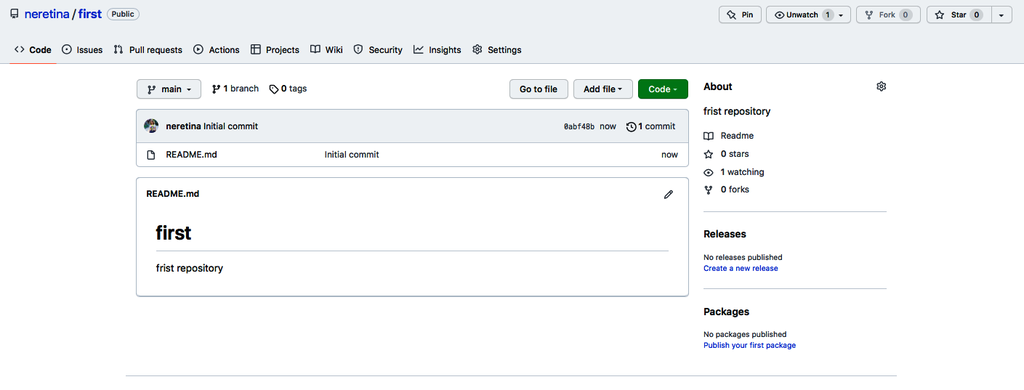
В поле Repository name введите уникальное название. В учётной записи не может быть двух одинаковых. Можно добавить краткое описание. А далее выбрать, каким будет репозиторий: общедоступным или приватным.

Настройки созданного репозитория

Если необходимо, выберите Add a README file. Это визуальное отображение информации о репозитории: разработчики объясняют примеры использования кода, зачем он нужен.

Визуальное отображение информации о репозитории

Финальный шаг для создания репозитория — выберите тип лицензии и нажмите кнопку Create repository.



Финальный шаг создания репозитория

Работа с ветками

По умолчанию в хранилище создаётся ветвь с именем main — это основная ветка. Дополнительные ветви используют для одновременного создания различных версий проекта. Это удобно, чтобы добавлять новые возможности, не изменяя основной исходный код. Можно использовать ветви для экспериментов и внесения правок до внесения изменений в основной код. Например, когда идёт работа над новыми фичами.

Когда создаётся ответвление от основной ветви, создаётся копия основной ветви в том виде, в котором она была в этот момент. Если кто-то другой вносил изменения в основную ветку, пока владелец ветки работал над своей, он сможет использовать обновления. Этого не происходит автоматически. Человек должен сам обновить свою ветку, если захочет.

Для создания ветви перейдите на вкладку «Code» в созданном репозитории и щёлкните на выпадающем списке веток и тегов в верхней части списка с надписью main. В текстовом поле введите название ветви, например edits, после чего нажмите «Create branch» — edits отразится в основной ветви.

Создание ветви

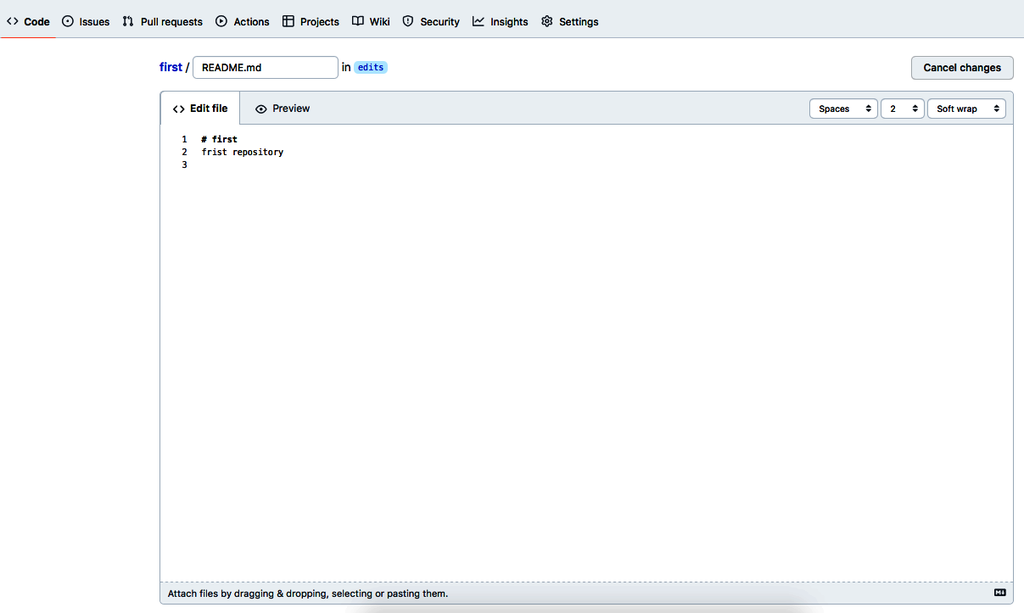
Теперь появились две ветви — основная и edits, которые выглядят одинаково. Добавлять изменения можно в новую ветку.

**Изменение файлов и коммиты**

В Git и впоследствии на GitHub сохранённые изменения в файлах репозитория называются коммитами. Каждый коммит имеет связанное с ним сообщение коммита — описание, объясняющее, почему сделано изменение. Сообщения коммита фиксируют историю изменений для других участников. Важно написать, что вы конкретно изменили.

Если пользователь сделал коммит на своём компьютере, то сразу же в центральном репозитории коммит не появится. Чтобы он там появился, необходимо выполнить команду push — надо отправить свою текущую версию ветки.

Для того чтобы сделать коммит, в созданной ветке edits отредактируйте файл README.md. В поле «Edit file» напишите сообщение об изменениях и сохраните их.



Чтобы сделать коммит, необходимо отредактировать файл README.md

● Таким образом, работа происходит так: Разработчик создаёт новую ветку репозитория, работает в ней, после чего делает коммит и пушит.

● Создаёт пулл реквест (pull request).

● Ревьюер перепроверяет работу и её одобряет, после чего объединяет с основной веткой.

**Отчёты об ошибках**

Issues позволяют отслеживать работу на GitHub, где происходит разработка. Можно выбрать наиболее удобный метод для рабочего процесса.

Например, вы можете создать отчёт об ошибках из репозитория, элемента в списке задач, заметки в проекте, комментария в проблеме, конкретной строке кода.

С тем, чтобы отслеживать отчёты об ошибках, помогут списки задач. А чтобы распределить связанные вопросы по категориям, удобно использовать метки.

**Управление проектами**

Репозиторий — это хранилище. Пользователь организует работу над этим кодом и видит доску с завершёнными задачами. Разберёмся, как выложить проект на GitHub.

Чтобы перейти во вкладку создания проекта, нажмите на аватар профиля в правом верхнем углу и выберите соответствующий пункт. Откроется меню, где отображаются все созданные проекты.

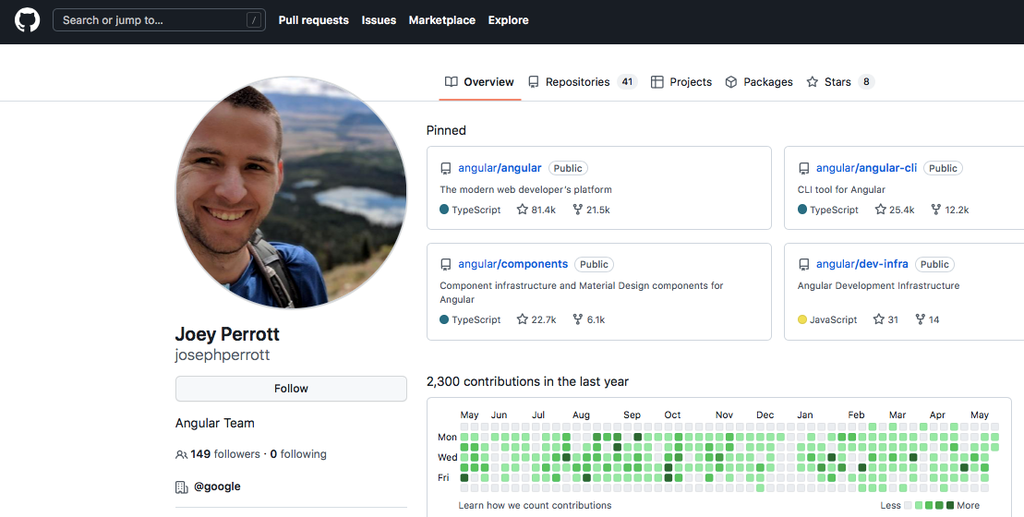
Настройка проекта на GitHub

Проект — это настраиваемая электронная таблица, которая интегрируется с ошибками и запросами на доработку на GitHub. Проекты гибкие, чтобы команды смогли работать с ними так, как им удобно, и добавлять пользовательские поля для отслеживания метаданных.

Проект автоматически обновляется в соответствии с информацией на GitHub. При изменении запроса на доработку проект отражает эти изменения.

GitHub — это и iCloud, и соцсеть для программистов и разработчиков приложений. Одна из основных целей GitHub — быть единым удобным местом для проектов с исходным кодом. Кроме этого, у каждого разработчика есть собственная страница, где можно посмотреть его активность.

Процесс участия в разработке чужих проектов с исходным кодом и заведение задач (багов) называется «контрибьютить» (contribute). Если человек, помимо своей основной работы, находит время на это, он вносит вклад в общее дело.



Знаменитая плиточка, где видна активность: отсылку к ней можно встретить в мемах

Чем известнее проект, в который разработчик контрибьютит, тем круче. Это одна из возможностей для начинающих программистов попробовать себя в реальной разработке, если найти работу сразу не получается, и показать примеры на собеседовании.