5 ПРАКТИЧЕСКАЯ РАБОТА

5.1 Цель работы: изучение принципов работы с системами контроля версий

5.2 Задачи:

- научиться создавать репозитории на GitHub;

- научиться работать с файлами проекта с помощью средств Git;

- научиться пользоваться терминалом Git.

5.3 Теоретические сведения

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. 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.

Если вы хотите работать на GitHub, то вовсе не обязательно быть гуру в программировании, ведь все самое основное делается прямо на сайте.

Не лишним будет разобраться с терминалом, поскольку терминальные команды действительно упрощают жизнь.

Если вы видите команду с угловыми скобками: < > , то смело удаляйте эти скобки и меняйте их содержимое на нужный вам текст.

Пример: `git add <имя_файла>`. Здесь вы можете написать нечто подобное: `git add hello_world.py` . Это означает, что вы хотите добавить в репозиторий файл под названием `hello_world.py`.

Для начала необходимо запомнить следующие терминальные команды:

```
git clone
git status
git add
git commit -m " "
git push
```

Затем к ним добавим еще вот эти:

```
git init
```

```
git branch
git merge
git checkout
```

Эти команды вам пригодятся в случае, если вы будете работать с другими людьми или захотите внести какие-то изменения в проект и протестировать их до создания коммита.

Не лишней будет и вот такая команда:

```
git help
```

Шаг 1: Регистрация и установка

Зайдите на GitHub и создайте свой аккаунт. В принципе, этим можно и ограничиться. При желании можете установить Git. Но для работы с GitHub это вовсе не обязательно. Однако если вы планируете заниматься проектами на локальном компьютере, то установка вам все-таки нужна.

Теперь перейдите в терминал, и начнем работу. Если хотите задать одно имя пользователя для **всех репозиториев** на компьютере, то напишите:

```
git config – global user.name "<ваше_имя>"
```

замените `<ваше_имя>` на свое имя в кавычках. Можете написать все, что угодно. Если хотите задать имя только для одного репозитория, то удалите из команды слово `global`.

Теперь напишите свой адрес электронной почты. Проследите, чтобы он совпадал с адресом, указанным при регистрации на GitHub.

```
git config – global user.email "<адрес_почты@email.com>"
```

Теперь вы готовы к работе с Git на локальном компьютере.

Начнем с создания нового репозитория на сайте GitHub. Вы также можете выполнить `git init` и создать новый репозиторий из директории проекта.

Репозиторий состоит из трех «деревьев». Первое «дерево» – это **рабочая директория,** в которой хранятся актуальные файлы. Второе – это **index** или область подготовленных файлов. А еще есть **head** – указатель на ваш последний коммит.

Вариант 1. Для опытных пользователей

Вот как начать работу с Git из терминала.

Если у вас есть директория проекта, то просто перейдите в терминал, а в самой директории проекта выполните команду

```
git init
```

Если хотите инициализировать проект со всеми файлами из директории проекта, то выполните команду

```
git init
```

Допустим, в вашем проекте есть папка `new_project`. Вы можете перейти в нее из окна терминала и добавить локальный репозиторий. Это делается через следующую команду:

```
cd new_project
git init
```

В вашем проекте появилась новая скрытая директория с названием `.git`. Именно здесь Git хранит все, что ему нужно для отслеживания проекта. Теперь вы можете последовательно добавлять файлы в область подготовки:

```
git add <имя_первого_файла>
```

или добавьте сразу все файлы через:

```
git add .
```

Создать коммит с этими изменениями можно через команду:

```
git commit -m "<сообщение_коммита>"
```

Если изменения вас устраивают, напишите:

```
git push
```

и отправьте эти изменения в репозиторий. Проверить, есть ли изменения для отправки, можно в любое время по команде:

```
git status
```

При внесении изменений следует обновить и сами файлы:

```
git add <имя_файла>
```

или

```
git add – all
```

Создайте коммит, добавьте нужное сообщение и отправьте этот коммит в репозиторий.

Вот и все! Теперь вы можете инициализировать репозиторий, создавать коммиты с файлами и сообщениями, а также отправлять коммиты в ветку `master`.

Вариант 2. Для начинающих пользователей

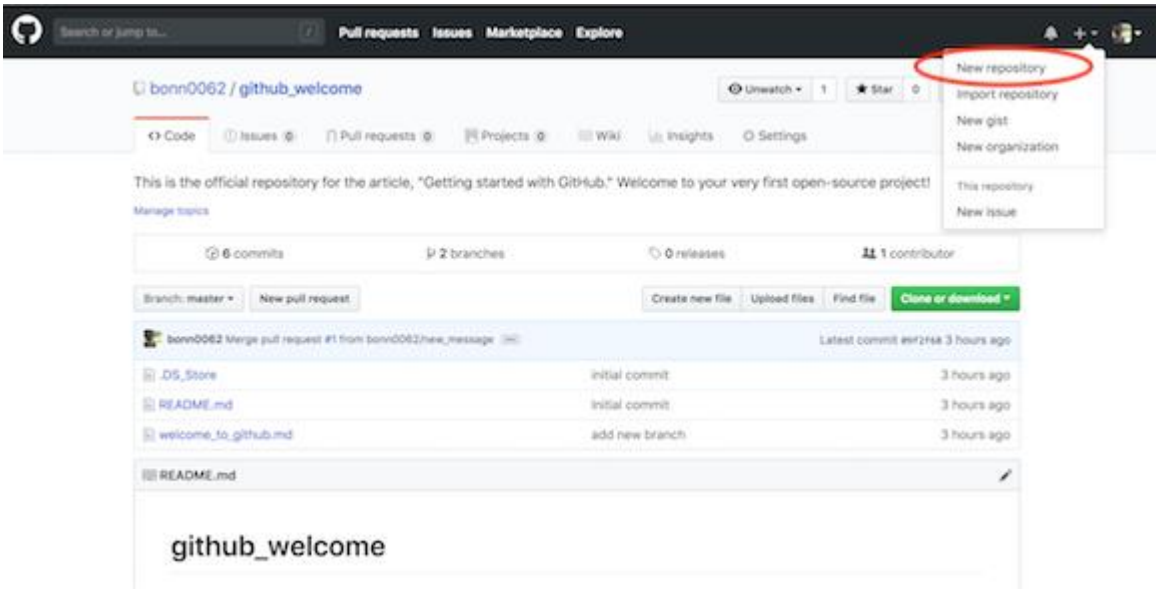
Этот вариант выбирают совсем новички в разработке. Вполне возможно, у вас уже есть целая папка с файлами проекта для размещения на GitHub, но вы не знаете, с чего начать.

Допустим, вы хотите создать новый репозиторий. Это место, где будет «жить » ваш проект. Если вы не хотите создавать новый репозиторий, то можете склонировать уже существующий. Именно так вы копируете чужой проект или берете нужную вам информацию для работы/учебы. Мы еще к этому вернемся, но чуть позже.

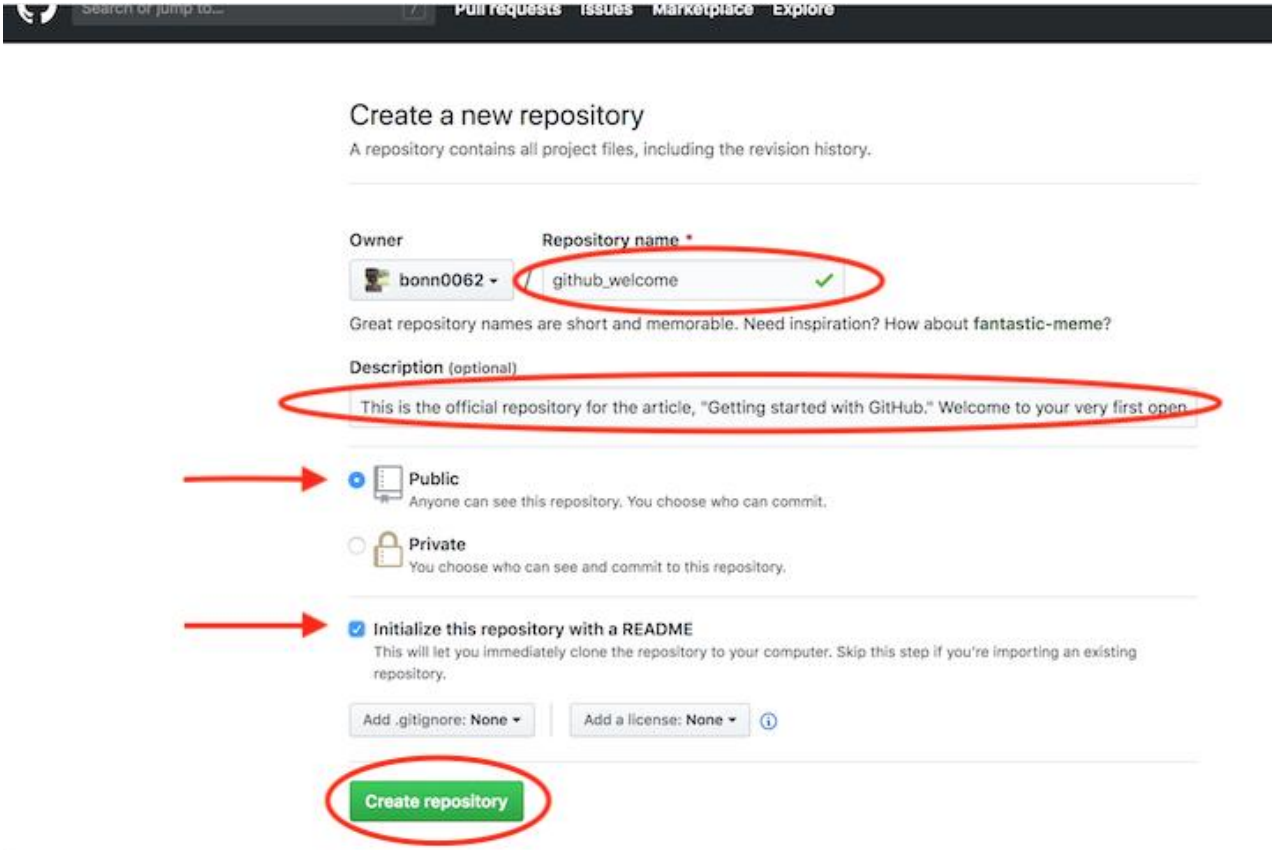
Репозиторий – это место, в котором вы систематизируете свой проект. Здесь вы храните файлы, папки, видео, изображения, блокноты Jupyter Notebook, наборы данных и т.д. Перед началом работы с Git необходимо инициализировать репозиторий для проекта и правильно его подготовить. Это можно сделать на сайте GitHub.

Лучше сразу добавлять в репозиторий **README**-файл с информацией о проекте. Это можно сделать в момент создания репозитория, поставив галочку в соответствующем поле.

- Перейдите на сайт GitHub. Нажмите на значок **+** в верхнем правом углу, а затем выберите **New repository**.
- Придумайте имя репозитория и добавьте короткое описание.
- Решите, будет ли этот репозиторий размещаться в открытом доступе или останется закрытым для просмотра.
- Нажмите **Initialize this repository with a README** для добавления README-файла. Настоятельно рекомендую снабжать все ваши проекты файлом-описанием, ведь README – это первая вещь, на которую люди обращают внимание при просмотре репозитория. К тому же, здесь можно разместить нужную информацию для понимания или запуска проекта.



Новый репозиторий



Создание нового репозитория

При желании можете уже сейчас начинать работать над проектом. Добавляйте файлы, вносите в них изменения и т.д. напрямую с сайта GitHub. Однако конечный результат подобной деятельности может вас немного огорчить.

Вносить изменения в проект можно двумя способами. Вы можете изменять файлы/блокноты на компьютере либо делать это на сайте GitHub.

Допустим, вам захотелось подкорректировать README-файл на сайте GitHub.

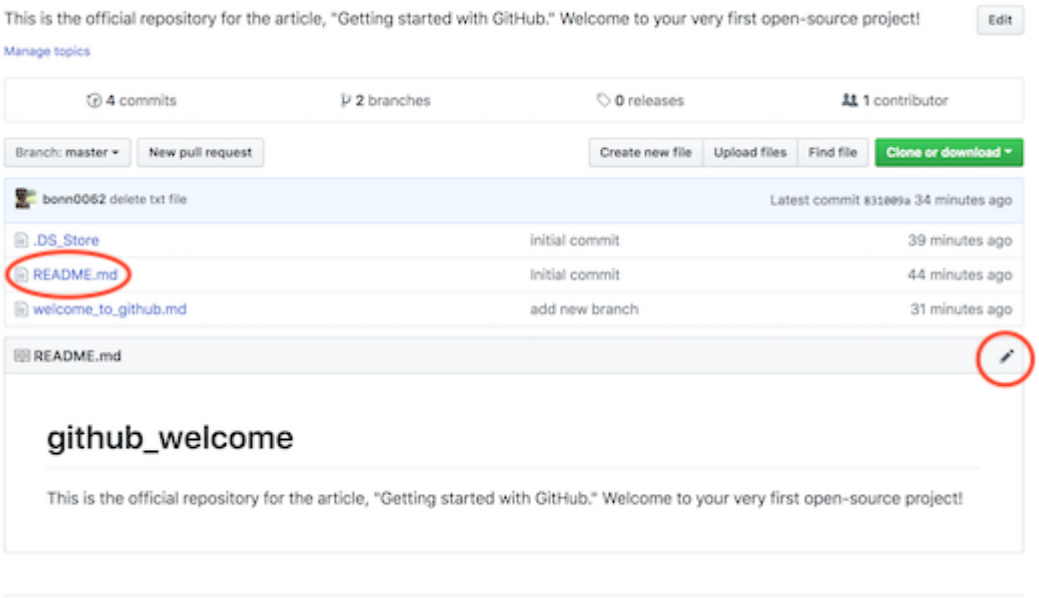
– Для начала перейдите в ваш репозиторий.

– Для выбора файла кликните по его названию (например, кликните по **README.md** для перехода к файлу-описанию).

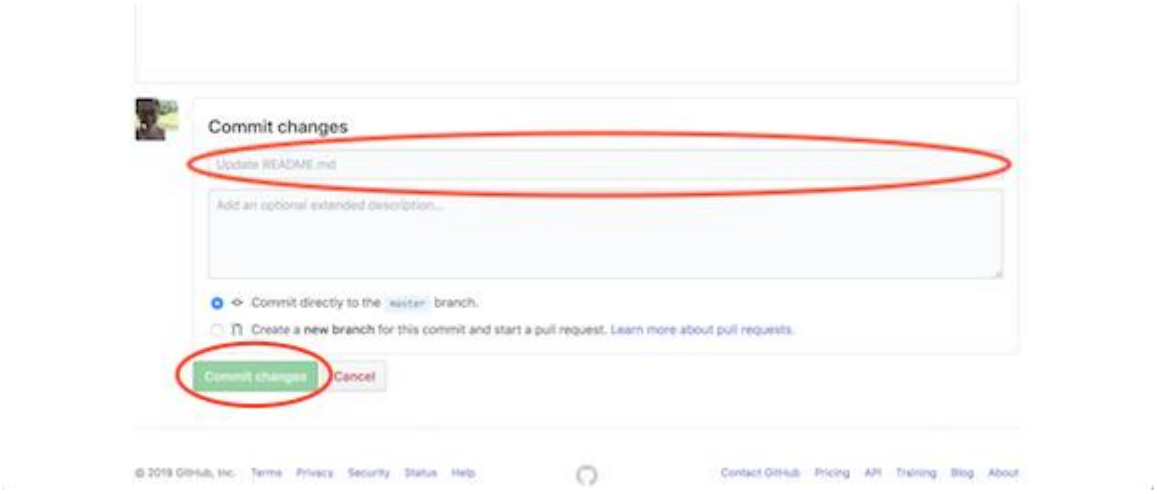
– В верхнем правом углу вы увидите иконку с карандашом. Нажмите на нее для внесения изменений.

– Напишите короткое сообщение, передающее суть изменений (и подробное описание, если сочтете это нужным).

– Нажмите кнопку **Commit changes**.



Изменение файла на GitHub



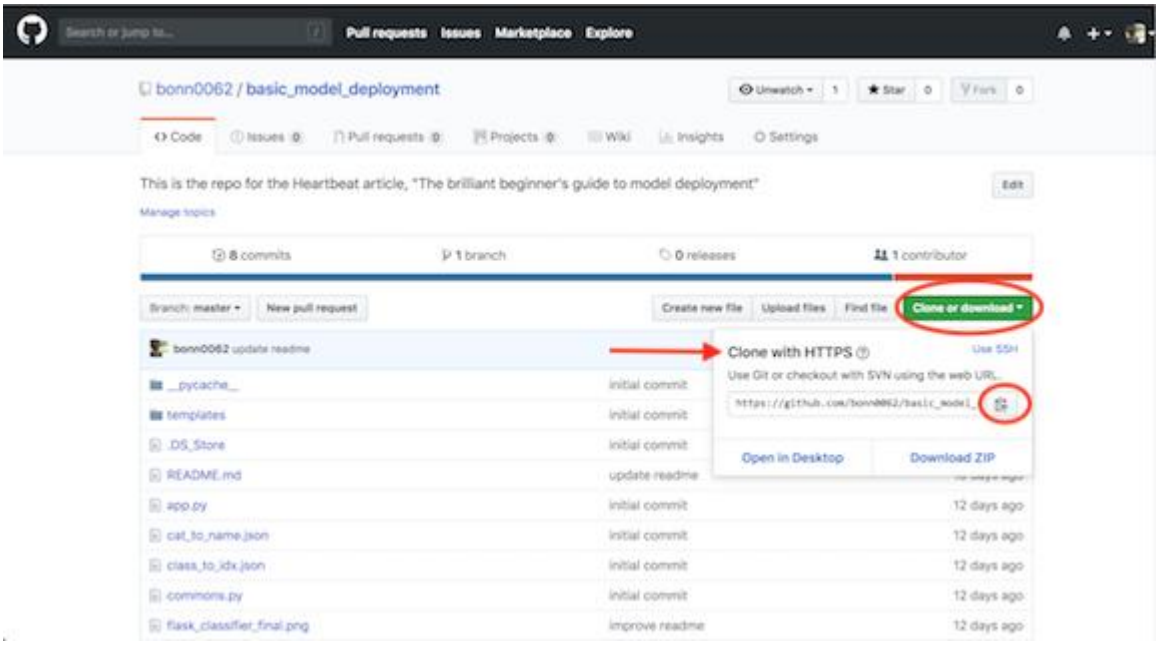
Подготовка коммита с изменениями

Вы успешно внесли изменения в README-файл своего нового репозитория! Обратите внимание на небольшую кнопку на картинке выше. Она позволяет создавать новую ветку этого коммита и добавлять Pull request. Запомните ее, скоро к ней вернемся.

Клонирование проектов / репозиториев

Возможно, вы захотите клонировать свой новый репозиторий для дальнейшей работы с ним на локальном компьютере. Либо у вас уже есть существующий репозиторий, который вы хотели бы клонировать.

Для **клонирования репозитория** на компьютер перейдите в репозиторий на GitHub и нажмите большую зеленую кнопку под названием **Clone or download** (разумеется, вы можете просто скачать репозиторий). Проследите, чтобы появилась надпись **Clone with HTTPS**. Теперь нажмите на иконку буфера обмена для копирования-вставки (либо выделите ссылку и скопируйте ее).



Клонирование или скачивание репозитория

Откройте **терминал** и перейдите в директорию для копирования репозитория. Например, для перехода на **Рабочий стол** напечатайте вот это:

```
cd Desktop
```

Затем клонируйте туда репозиторий по следующей команде:

```
git clone <то,_что_вы_только_что_скопировали>
```

Не забудьте изменить информацию в угловых скобках на нужную вам. И удалите сами скобки < >.

Если вы не очень хорошо ориентируетесь в терминале, то переход по директориям можно осуществлять через команду `cd`. Например, откройте терминал и напечатайте `ls` для отображения перечня доступных директорий. Вполне возможно, что в этом списке вы сразу увидите директорию `Desktop`. Либо напечатайте `cd Desktop`. Далее выполните команду `git clone` и склонируйте репозиторий на Рабочий стол.

Бывает и так, что вместо перечня расположений, вы видите различные имена пользователей. Тогда до того, как перейти в `Desktop`, вам потребуется выбрать нужного пользователя через команду `cd <пользователь>` (замените `<пользователь>` на нужное вам имя). Затем снова напечатайте `ls`, чтобы увидеть весь список. И вот теперь, увидев в списке `Desktop`, смело печатайте `cd Desktop`. Сейчас уже можно выполнять `git clone`!

Если вдруг в терминале вы захотите «откатиться» на шаг назад, то напишите `cd ..`

Новый GitHub-репозиторий, склонированный на рабочий стол, готов! Данная команда создает точную копию репозитория в вашей системе. Здесь вы сможете с ним работать, редактировать, индексировать изменения, создавать коммиты с изменениями и отправлять их на GitHub.

Совсем не обязательно создавать репозиторий на Рабочем столе. Клонировать можно в любое место на компьютере. Команду `git clone` можно выполнять и сразу после открытия терминала. Однако, если вы не очень любите копаться в папках на компьютере, то неплохо будет разместить проект на виду, то есть на Рабочем столе…

Если хотите просто покопаться в каком-то проекте, то вместо клонирования можете сделать **форк** проекта на GitHub. Для этого нажмите кнопку **Fork** в верхнем правом углу сайта. Так вы добавите копию этого проекта в свои репозитории и сможете вносить туда любые изменения без вреда для оригинала.

Добавляем файлы в проект

Вот, чем мы займемся:

```
git status
git add
git commit -m " "
git push
```

Должно быть, у вас уже есть файлы, которые вы бы хотели разместить в новом репозитории. Отыщите их на компьютере и перетащите в новую папку репозитория на Рабочем столе.

Проверьте **статус** проекта.

Откройте терминал и перейдите в папку репозитория. Для проверки обновлений выполните:

```
git status
```

Если вы перетаскивали файлы в папку проекта, то потребуется обновить состояние репозитория. Добавлять файлы в репозиторий можно по одному:

```
git add <имя_файла>
```

Либо все сразу:

```
git add – all
```

или даже:

```
git add .
```

Это ваши предлагаемые изменения. Операцию можно повторить с новыми файлами либо с уже существующими, но измененными. По сути, ничего нового в сам проект вы не добавляете. Вы всего лишь загружаете новые файлы и указываете Git на эти изменения.

Процесс создания коммитов с изменениями начинается с выполнения команды:

```
git commit -m "<сообщение_о_коммите>"
```

Коммиты изменений добавляются в **head** (указатель), а не в удаленный репозиторий. Не забудьте заменить текст в скобках и убрать <>. После внесения изменений создается снимок состояния репозитория, для чего используется команда`commit`. А через `-m` добавляется сообщение об этом снимке.

Сохраненные изменения и называются коммитом. При создании коммита вы добавляете сообщение о том, что именно менялось и почему. Так другие люди смогут лучше понять суть изменений.

Теперь ваши изменения сохранены в указателе локальной копии проекта. Для отправки изменений на удаленный репозиторий выполните команду:

```
git push
```

Тем самым вы отправляете изменения напрямую в репозиторий. Если вы работаете на локальном компьютере и хотите, чтобы коммиты отображались в онлайн, то необходимо своевременно отправлять эти изменения на GitHub по команде `git push`.

Актуальность версии можно проверить в любое время через команду `git status`.

Итог: у вас есть свой GitHub репозиторий, вы научились добавлять и изменять в нем файлы.

5.4 Ход работы

1) Изучить основные теоретические сведения к работе

2) Установить на рабочем компьютере GitHub Desktope и Git for Windows (Git for Linux)

3) Пройти регистрацию на GitHub

4) Создать GitHub-репозиторий для собственных проектов

5) Добавить файлы проекта в созданный репозиторий (можно использовать файлы, полученные в результате выполнения лабораторных работ по дисциплине Основы алгоритмизации и программирования)

6) Подготовить отчет о проделанной работе