Лабораторная работа №2

Рогозин Игорь Андреевич

Содержание

# Цель работы

Целью работы является освоение процедуры оформления отчетов с помощью легковесного языка разметки Markdown.

# Задание

1. Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs>lab02>report).
2. Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.
3. Загрузите файлы на github.

# Теоретическое введение

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения,сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial.

Понятия в VCS: Хранилище – это система, которая позволяет хранить все версии существовавших файлов. Commit – запись изменений. История – список изменений раннее. Рабочая копия – копия файла, с которой активно ведется работа. В классических системах контроля версий используется централизованная модель,предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Децентрализованная VCS предназначена для обработки ветвлений и слияний как части своей ДНК, сохраняя указатель на предыдущий коммит в каждом коммите, поэтому любое изменение можно отследить до общего предка. Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

# Выполнение лабораторной работы

# Начало

1. Откройте терминал

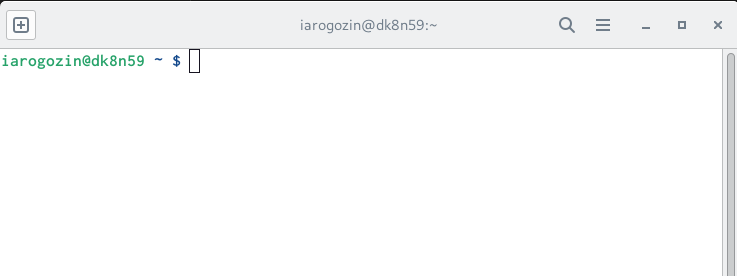


Рис.1 Терминал линукс

1. После создания аккаунта на сайте [link text](https://github.com/) сделаем предварительную конфигурацию git. Вводим в консоль команды, указав имя и email владельца репозитория:

Рис. 2 Настройка конфигурации git

Рис. 2 Настройка конфигурации git

1. Настроим utf-8 в выводе сообщений git:

Рис. 3 Настройка конфигурации git

Рис. 3 Настройка конфигурации git

1. Зададим имя начальной ветки (будем называть её master):

Рис. 4 Настройка конфигурации git

Рис. 4 Настройка конфигурации git

1. Зададим параметры: Параметр autocrlf: Параметр safecrlf:

Рис. 5 Настройка конфигурации git

Рис. 5 Настройка конфигурации git

# Создание SSH ключа

1. Теперь для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый):

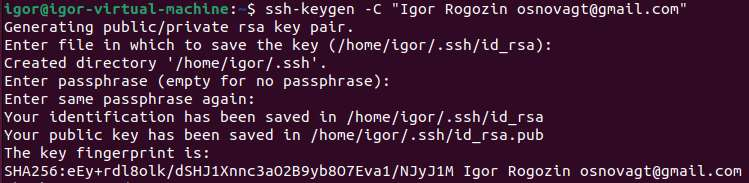
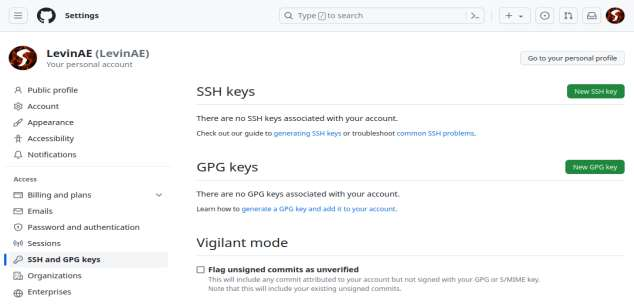


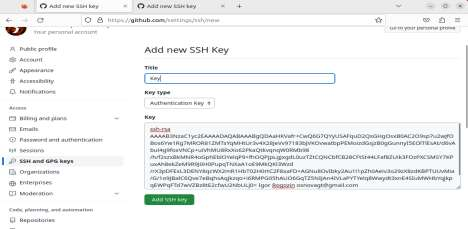
Рис. 2.1: Создание SSH-ключа в терминале

1. Далее загрузим сгенерированный ключ на github. Для этого зайти на сайт http://github.com/ под своей учётной записью и перейти в меню Settings. После этого выбрать в боковом меню SSH and GPG keys. Перейдём в подкаталог Документы домашнего каталога указав относительный путь:



Риc. 2.2: Настройки ключей github

1. Копируем ключ с помощью команды cat ~/.ssh/id\_rsa.pub | xclip -sel clip и вставляем ключ в появившееся на сайте поле и указываем для ключа имя (Title). Нажимаем кнопку Add SSH Key

Рис. 2.3: Сохранение ключа на github 

# Создание рабочего пространства и репозитория курса на основе шаблона

1. Откроем терминал и создадим каталог для предмета «Архитектура компьютера»:

Рис. 3.1: Создание основной директории

Рис. 3.1: Создание основной директории

Репозиторий на основе шаблона можно создать через web- интерфейс github.

1. Перейдём на станицу репозитория с шаблоном курса [link text](https://github.com/yamadharma/cour%20se-directory-student-template). Далее выберите Use this template.
2. В открывшемся окне задайте имя репозитория (Repository name) study\_2023–2024\_arhpc и создайте репозиторий (кнопка Create repository from template).

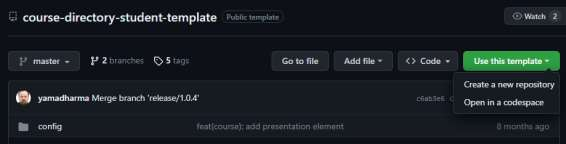


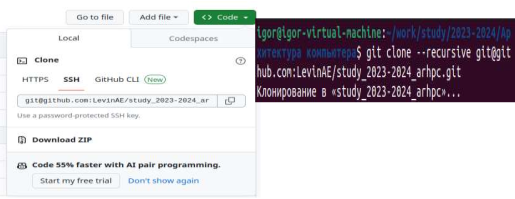
Рис. 3.2: Создание основной директории на github



Рис. 3.3: Создание основной директории на github

1. Теперь зайдём в терминал и перейдём в каталог курса:



И клонируем созданный репозиторий: 

# Настройка каталога курса

1. Для начала перейдём в каталог курса:

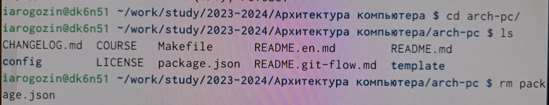


Рис. 4.1: Удаление файла package.json

1. Создадим необходимые каталоги:

Рис. 4.2: Создание каталогов и проверка

Рис. 4.2: Создание каталогов и проверка

1. После этого отправим все изменения на сервер

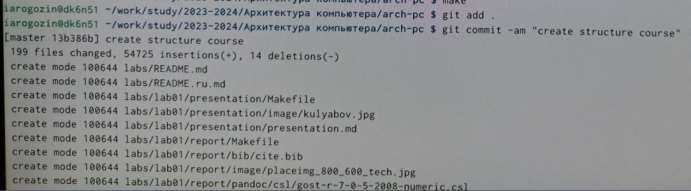


Рис. 4.3 Отправка файлов на github

# Задание для самостоятельной работы

1. Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs>lab02>report).

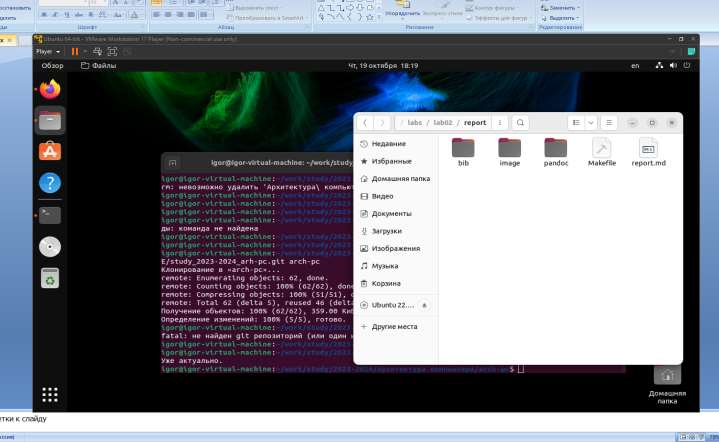


Рис. 1c

1. Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.
2. Загрузите файлы на github

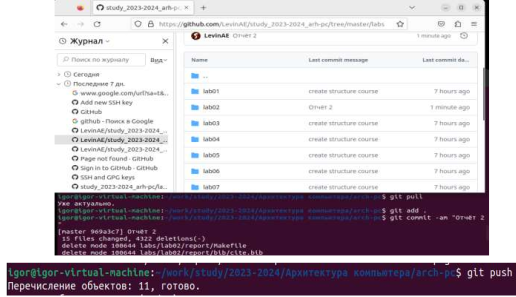


Рис. 3c

# Выводы

Сегодня я научился работать с github и сохранять, а затем загружать отчёты в репозиторий.