Отчёт по лабораторной работе №2

Дисциплина: архитектура компьютера

Закиров Нурислам Дамирович

Содержание

Список иллюстраций

1 Цель работы

Цель этой работы состоит в том, чтобы изучить идеологию и использование инструментов контроля версий, а также получить практические навыки работы с системой git.

2 Задание

1. Настройка GitHub.
2. Базовая настройка Git.
3. Создание SSH-ключа.
4. Создание рабочего пространства и репозитория курса на основе шаблона.
5. Создание репозитория курса на основе шаблона.
6. Настройка каталога курса.
7. Выполнение заданий для самостоятельной работы.

3 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. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить изменения, сделанные разными участниками, 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. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд. 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. Работа пользователя со своей веткой начинается с проверки и получения изменений из центрального репозитория (при этом в локальное дерево до начала этой процедуры не должно было вноситься изменений). Затем можно вносить изменения в локальном дереве и/или ветке. После завершения внесения какого-то изменения в файлы и/или каталоги проекта необходимо разместить их в центральном репозитории.

4 Выполнение лабораторной работы

4.1 Настройка GitHub

Создаю учетную запись на сайте GitHub. Аккаунт создан (рис. [**??**]).

Аккаунт GitHub

4.2 Базовая настройка Git

Запускаем терминал, после чего делаю предварительную конфигурацию git. Ввожу команду git config –global user.name , указывая свое имя и команду git config –global user.email “work@mail”, указывая в ней мою электронную почту (рис. [4.1]).

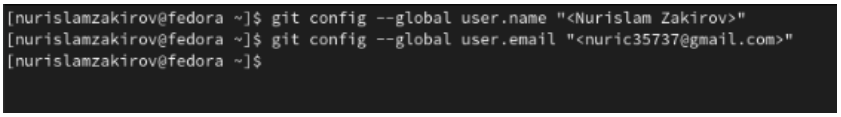

Рис. 4.1: Предварительная конфигурация git

Настраиваю git для корректного отображения символов (рис. [4.2]).

Рис. 4.2: Настройка кодировки

Задаю имя «master» для начальной ветки (рис. [4.3]).

Рис. 4.3: Создание имени для начальной ветки

Даём параметр autocrlf со значением input (рис. [4.4]).

Рис. 4.4: Параметр autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость (рис. [4.5]). При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации.

Рис. 4.5: Параметр safecrlf

4.3 Создание SSH-ключа

Ввожу команду ssh-keygen -C “Имя Фамилия, work@email”, указывая имя владельца и электронную почту владельца (рис. [**??**]). Ключ автоматически сохранится в каталоге ~/.ssh/.

Генерация SSH-ключа

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip (рис. [**??**]).

Копирование содержимого файла

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key» (рис. [**??**]).

Окно SSH and GPG keys

Вставляю скопированный ключ в поле «Key». Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис. [**??**]).

Добавление ключа

4.4 Создание рабочего пространства и репозитория курса на основе шаблона

Я закрываю браузер и запускаю терминал. Создаю директорию и рабочее пространство с помощью утилиты mkdir, используя ключ -p после домашней директории ~/work/study/2022-2023/Computer Architecture. Далее я проверяю, были ли созданы необходимые мне каталоги с помощью ls (рис. [**??**]).

Создание рабочего пространства

4.5 Создание репозитория курса на основе шаблона

Заходим на страницу репозитория с шаблоном курса по адресу https://github.com/yamadharma directory-student-template. Выбираем «Use this template», чтобы использовать этот шаблон для своего репозитория (рис. [**??**]).

Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): study_2023–2024_arh-pc и создаю его, нажав кнопку «Create repository from template» (рис. [**??**]).

Окно создания репозитория

Репозиторий создан (рис. [**??**]).

Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты cd (рис. [**??**]).

Перемещение между директориями

Клонирую созданный репозиторий с помощью команды git clone –recursive git@github.com:/study_2022–2023_arh-pc.git arch-pc (рис. [**??**]).

Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. [**??**]).

Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd (рис. [**??**]).

Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm (рис. [4.6]).

```
[nurislamzakirov@fedora arch-pc]$ rm package.json
[nurislamzakirov@fedora arch-pc]$
```

Рис. 4.6: Удаление файлов

Создаю необходимые каталоги (рис. [4.7]).

```
[nurislamzakirov@fedora arch-pc]$ echo arch-pc > COURSE
[nurislamzakirov@fedora arch-pc]$ make
```

Рис. 4.7: Создание каталогов

Добавляю все созданные каталоги с помощью git add, комментирую и сохраняю изменения на сервере как добавленипше курса с помощью git commit (рис. [**??**]).

Добавление и сохранение изменений на сервере

Отправляю все на сервер с помощью push (рис. [**??**]).

Выгрузка изменений на сервер

Проверяю верность выполнения работы сначала на самом сайте GitHub (рис. [**??**]).

Страница репозитория

4.7 Выполнение заданий для самостоятельной работы

1. 1. Перехожу в директорию labs/lab02/report с помощью утилиты cd. Создаю в каталоге файл для отчета по второй лабораторной работе с помощью утилиты touch (рис. [**??**]).

Создание файла

Перемещаю первую лабораторную работу в /home/nurislamzakirov/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report , после чего перемещаюсь в терминале по этому же пути и добавляю данный файл в репозиторий с помощью git add, а затем сохраняю изменения с помощью git commit, и в конце отправляю в центральный репозиторий сохраненные изменения командой git push -f origin master (рис. [**??**]).

Добавление файла в репозиторий

Проверяю на сайте GitHub верность пройденных действий. Мы видим, что теперь в нашем репозитории добавились отчеты по выполнению первой и второй лабораторной работе (рис. [**??**], рис. [**??**]).

Страница каталога в репозитории

Страница каталога в репозитории

5 Выводы

При выполнении данной лабораторной работы я изучил идеологию и применение средств контроля версий, а также приобрел практические навыки по работе с системой git. # Список литературы

1. Архитектура ЭВМ
2. Git - gitattributes Документация