ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Архитектура компьютера

Лисенков Е.Р

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Изучить идеологию и применение средств контроля версиями, а также получить практические навыки по работе 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 шаблоа.
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 п роекта 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, произ ведённые разными участниками проекта, производить откат к любой более ранн ей версии проекта, если это требуется. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Вы полнение большинства функций по управлению версиями осуществляется специа льным сервером. Участник проекта (пользователь) перед началом работы посред ством определённых команд получает нужную ему версию файлов. После внесе ния изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно в ернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохра нять только изменения между последовательными версиями, что позволяет умен ьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и раз решения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными у частниками (автоматически или вручную), вручную выбрать нужную версию, от менить изменения вовсе или заблокировать файлы для изменения. В зави-

симости от настроек блокировка не позволя ет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким об разом, привилегированный доступ только одному пользователю, работающему с файлом. 6 Системы контроля версий также могут обеспечивать дополнительные, боле е гибкие функциональные возможности. Например, они могут поддерживать раб оту с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. К роме того, обычно доступна информация о том, кто из участников, когда и как ие изменения вносил. Обычно такого рода информация хранится в журнале изм енений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий ц ентральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди рас пределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд. Система контроля версий Git представляет собой набор программ коман дной строки. Доступ к ним можно получить из терминала посредством ввода ко манды git с различными опциями. Благодаря тому, что Git является распределённой системой контроля версий, рез ервную копию локального хранилища можно сделать простым копированием ил и архивацией. Система контроля версий Git представляет собой набор програм м командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Благодаря тому, что Git является рас пределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

4 Выполнение лабораторной работы

11.1 Настройка GitHub Первым делом создам учётную запись на сайте GitHub и всё заполню (рис. 4.1).

Рис. 4.1 Аккаунт GitHub 11.2 Базовая настройка GitHub После открытия виртуальной машины, начинаю настройку с помощью: git config –global user.name “” И git config –global user.email “work@mail” куда я ввожу свою почту (рис. 4.2).

Рис. 4.2 Конфигурация Git Далее настраиваю utf-8 в выводе сообщений git для отображения символов (рис. 4.3).

Рис. 4.3 Настройка кодировки Задаю имя «master» для начальной ветки (рис. 4.4).

Рис. 4.4 Создание имени для начальной ветки Задам параметр autocrlf со значением input, для того чтобы конвертировать CRLF в LF только при коммитах (рис. 4.5).

Рис. 4.5 Параметр autocrlf Задам параметр safeclf со значением warm, чтобы Git мог проверять преобразования на обратимость (рис. 4.6). 8

Рис. 4.6 Параметр safecrlf 11.3 Создание SSH-ключа Чтобы идентифицировать пользователя на сервере репозиториев нужно сделать пару ключей с помощью команд: ssh-keygen -C “Имя Фамилия, work@email” (рис. 4.7).

Рис. 4.7 Генерация SSH-ключа Далее установим утилиту Xclip, которая позволяет нам скопировать любой текст через терминал. На Linux мы должны скачать её, поэтому проделаем ко манды, которые вы увидите у меня на скриншотах (рис. 4.8 - рис. 4.11).

Рис. 4.8 Установка Xclip 9

Рис. 4.9 Установка Xclip

Рис. 4.10 Установка Xclip Делаю запрос на ключ SSH, который потребуется нам для GitHub (рис. 4.11).

Рис. 4.11 Запрос ключа SSH Перехожу на сайт GitHub и прикрепляю ключ для соединения с GitHub (рис. 4.12). 11.4 Создание рабочего пространства и репозитория курса на основе шаблона Закрываю браузер, открываю терминал. Создаю директорию, рабочее пространство, с помощью утилиты mkdir, блягодаря ключу -p создаю все директории после домашней ~/work/study/2022-2023/“Архитектура компьютера” рекурсивно. Далее проверяю с помощью ls, действительно ли были созданы необходимые мне каталоги (рис. 4.12).

Рис. 4.12 Создание рабочего пространства 10 11.5 Создание репозитория курса на основе шаблона Выполню переход по ссылке https://github.com/yamadharma/course-directorystudent-template и выбираю «Use this tamplate», чтобы использовать шаблон для своего репозитория (рис. 4.13).

Рис. 4.13 Страница шаблона В открывшемся окне нажимаю на кнопку «Create repository» (рис. 4.14).

Рис. 4.14 Создание репозитория Созданный репозиторий (рис. 4.15). 11

Рис. 4.15 Репозиторий создан Далее перехожу в созданный каталог курса с помощью cd (рис. 4.16).

Рис. 4.16 Переход в репозиторий Клонирую созданный репозиторий с помощью команды git clone –recursive git@github.com:/study_2022–2023_arh-pc.git arch-pc (рис. 4.17).

Рис. 4.17 Клонирование репозитория Копирую ссылку для клонирования на странице созданного репозитория (рис. 4.18).

Рис. 4.18 Окно с ссылкой для копирования репозитория 12 11.6 Настройка каталога курса Выполню переход в каталог arch-pc с помощью cd (рис. 4.19).

Рис. 4.19 Переход по директориям Удалю лишние файлы благодаря утилите rm (рис. 4.20).

Рис. 4.20 Удаление файлов Создаю необходимые каталоги (рис. 4.21).

Рис. 4.21 Создание каталогов Отправляю созданные каталоги с локального репозитория на сервер: добавил с помощью git add, комментирую, сохраняю изменения и добавляю на сервер с помощью команды git commit (рис. 4.22 - рис. 4.23).

Рис. 4.22 Загрузка на сервер 13

Рис. 4.23 Выгрузка на сервер Выполню проверку правильности выполнения работы на GitHub (рис. 4.24)

Рис. 4.24 Страница репозитрория 11.7 Выполнение заданий для самостоятельной работы 1) Выполню переход в директорию labs/lab03/report с помощью утилиты cd. Создам файл для отчёта с помощью утилиты touch (рис 4.25).

Рис. 4.25 Создание файла Оформить отчёт я могу в LibreOffice, поэтому открываю пустой лист без имени и в самом файле открываю недавно созданный отчёт в lab03/report (рис. 4.26).

Рис. 4.26 Открытый файл из lab03/report 14 2) Выполню переход в подкаталог lab01/report (рис. 4.27).

Рис. 4.27 Переход между каталогами Проверю местонахождения файлов с отчётами по лабораторным, с помощью команды ls (рис. 4.28).

Рис. 4.28 Проверка в директории Загрузки Копирую первую лабораторную и вторую с помощью утилиты cp и проверяю правильность выполнения команды cp с помощью ls (рис. 4.29).

Рис. 4.29 Копирование файлов Выполню проверку файлов и пойму, что сделал всё правильно (рис. 4.30).

Рис. 4.30 Нахождение файлов 3) Добавляю с помощью команды git add в коммит созданные файлы (рис. 4.31).

Рис. 4.31 Добавление на сервер 15 Перехожу в директорию к первой лабораторной и тоже добавляю её на сервер (рис. 4.32).

Рис. 4.32 Добавление на сервер 2-ого файла Сохраняю изменения на сервере и поясняю, что добавил файл (рис. 4.33).

Рис. 4.33 Сохраняю на сервере Отправляю в центральный репозиторий сохра-

ненные изменения командой git push -f origin master (рис. 4.34).

Рис. 4.34 Отправка в центральный репозиторий Проверяю на сайте GitHub правильность выполнения задания (рис. 4.35).

Рис. 4.35 Проверка выполненных команд

5 Выводы

Из сегодняшней лабораторной я понял как нужно работать с системой контроля GitHub. Эти знания понадобятся мне в дальнейшей работе, когда мне придётся работать в команде, где нужен будет контроль над файлами.

Список литературы

Лабораторная_р абота_№2_Систем