

Отчёт по лабораторной работе №2

Дисциплина: Архитектура компьютеров

Мантуров Татархан Бесланович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
	Список литературы	12

Список иллюстраций

4.1	Установка утилиты xslip.	10
4.2	Перемещение в каталог курса.	10
4.3	Удаление лишних файлов.	10
4.4	Создание необходимых каталогов.	10
4.5	Отправка файлов на сервер.	10
4.6	Создание файла.	11
4.7	Добавление файлов на сервер.	11

Список таблиц

1 Цель работы

Целью данной работы является изучение идеологии и применение средств контроля версий. Приобретение практических навыков 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 шаблона
5. Создание репозитория курса на основе шаблона
6. Настройка каталога курса
7. Задание 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 (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. 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. 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.

4 Выполнение лабораторной работы

Открываю терминал для дальнейшей работы. Выполняю предварительную конфигурацию git: открыв терминал ввожу команды `git config --global user.name ""` и `git config --global user.email "work@mail"`, указывая имя и email владельца репозитория. Выполнение предварительной конфигурации git

Настраиваю utf-8 в выводе сообщений git. Настройка кодировки

Задаю имя начальной ветки(будем называть ее master). Создание имени начальной ветки

Задаю параметр `autocrlf` и параметр `safecrlf` со значениями `input` и `warn` соответственно. параметры `autocrlf` и `safecrlf`

Создание SSH ключа

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый). Для этого использую команду `ssh-keygen -C "Имя Фамилия work@mail"`. Ключи сохраняются в каталоге `~/.ssh/`. Генерация SSH-ключа

Далее необходимо загрузить сгенерённый открытый ключ. Для этого зайти на сайт <http://github.org/> под своей учётной записью и перейти в меню Setting. После этого выбрать в боковом меню SSH and GPG keys и нажать кнопку New SSH key. Скопировав из локальной консоли ключ в буфер обмена при помощи команды `cat ~/.ssh/id_rsa.pub | xclip -sel clip` вставляю ключ в появившееся на сайте поле и указываем для ключа имя. Обратим внимание на команду, где `xclip` – утилита для копирования любого текста через терминал. Но изначально она не установлена в дистрибутиве, поэтому для дальнейшей работы устанавливаем

при помощи команды `apt-get install` с ключом `-y`, при этом в начале команды использую `sudo` (для установки от имени админа. (рис. ??))

Установка утилиты `xclip`.

Рис. 4.1: Установка утилиты `xclip`.

После установки утилиты `xclip` мы можем воспользоваться командой `cat ~/.ssh/id_rsa.pub | xclip -sel clip` Копирование содержимого файла.

Создаю директорию и рабочее пространство с помощью утилиты `mkdir`, благодаря ключу `-p` создаю все директории после домашней `~/work/study/2023-2024/` "Архитектура компьютера" рекурсивно. Создание рабочего пространства.

Открываю терминал и перехожу в каталог курса.

Перемещение в каталог курса.

Рис. 4.2: Перемещение в каталог курса.

Клонирую созданный репозиторий, предварительно копируя ссылку для клонирования, которую можно найти на странице созданного репозитория.

Перехожу в каталог курса при помощи `cd` и удаляю лишние файлы при помощи команды `rm package.json`. (рис. ??)

Удаление лишних файлов.

Рис. 4.3: Удаление лишних файлов.

Создаю необходимые каталоги(рис. ??). И отправляю файлы на сервер(рис. ??).

Создание необходимых каталогов.

Рис. 4.4: Создание необходимых каталогов.

Отправка файлов на сервер.

Рис. 4.5: Отправка файлов на сервер.

Выполнение заданий для самостоятельной работы.

Перехожу в директорию `labs/lab02/report` при помощи `cd` и создаю в каталоге файл для отчета по второй лабораторной работе.

Создание файла.

Рис. 4.6: Создание файла.

Добавляю файл отчет, используя `git add`. Сохраняю изменения на сервере при помощи команды `git commit -m`.

Добавление файлов на сервер.

Рис. 4.7: Добавление файлов на сервер.

#Выводы

В ходе лабораторной работы я изучила идеологию и применение средств контроля версий и приобрела практические навыки по работе с системой `git`.

Список литературы

1. Архитектура ЭВМ
2. Инструкция по использованию git