Лабораторная работа №2

Операционные системы

Саттарова Вита Викторовна

Содержание

List of Figures

1 Цели и задачи

1.1 Цель

Изучить идеологию и применение средств контроля версий.

1.2 Задачи

1. Установить необходимое ПО
2. Изучить информацию о системе контроля версий
3. Настроить репозиторий на Github

2 Объект и предмет исследования

2.1 Объект исследования

Системы контроля версий, Github.

2.2 Предмет исследования

Изучение особенностей систем контроля версий и работы с ними.

3 Условные обозначения и термины

Условные обозначения и термины отсутствуют

4 Теоретические вводные данные

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 дельтакомпрессию — 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. В зави-

симости 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. Кроме того, 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.

5 Техническое оснащение и выбранные методы проведения работы

5.1 Техническое оснащение

Персональный компьютер, интернет.

5.2 Методы

Анализ предложенной информации, выполнение работы по указанному алгоритму, получение дополнительной информации из интернета.

6 Выполнение лабораторной работы

1. Создала аккаунт на Github и ознакомилась с текстом работы. (рис. -fig. 6.1)

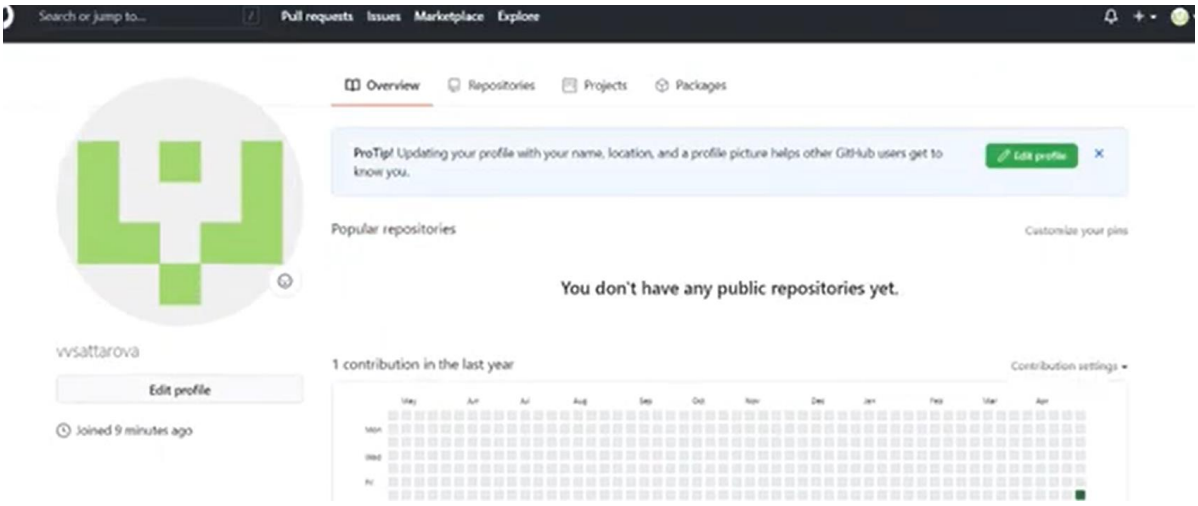

Figure 6.1: Рис. 1

2. Установила пакет Chocolately, с помощью которого установила пакет Git. (рис. -fig. 6.2)

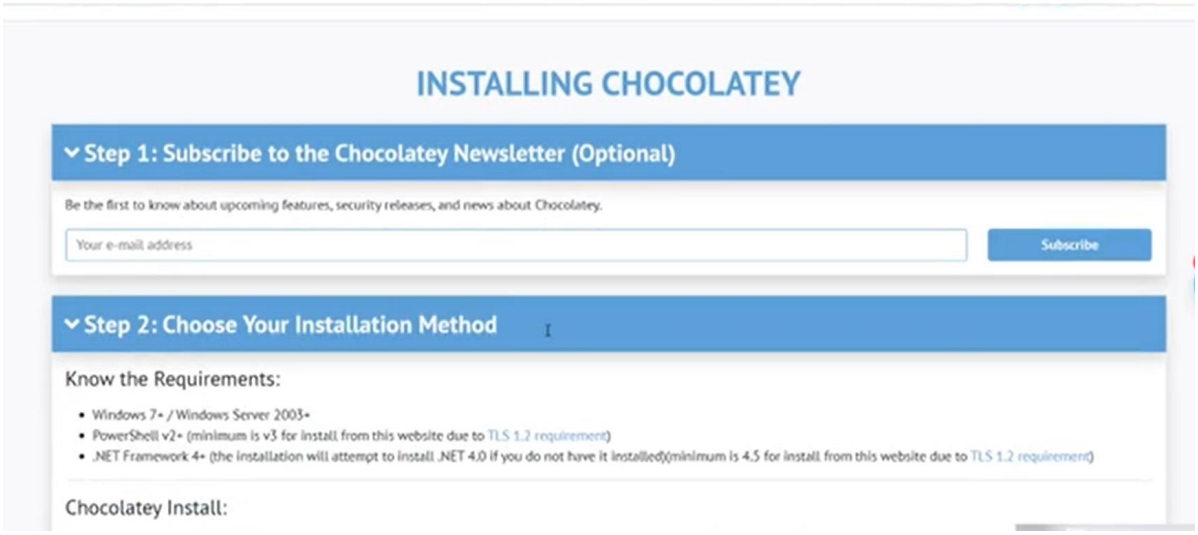

Figure 6.2: Рис. 2

3. Начала настройку Гитхаба, установила имя пользователя, электронную почту, добавила публичный ключ. (рис. -fig. 6.3) (рис. -fig. 6.4)

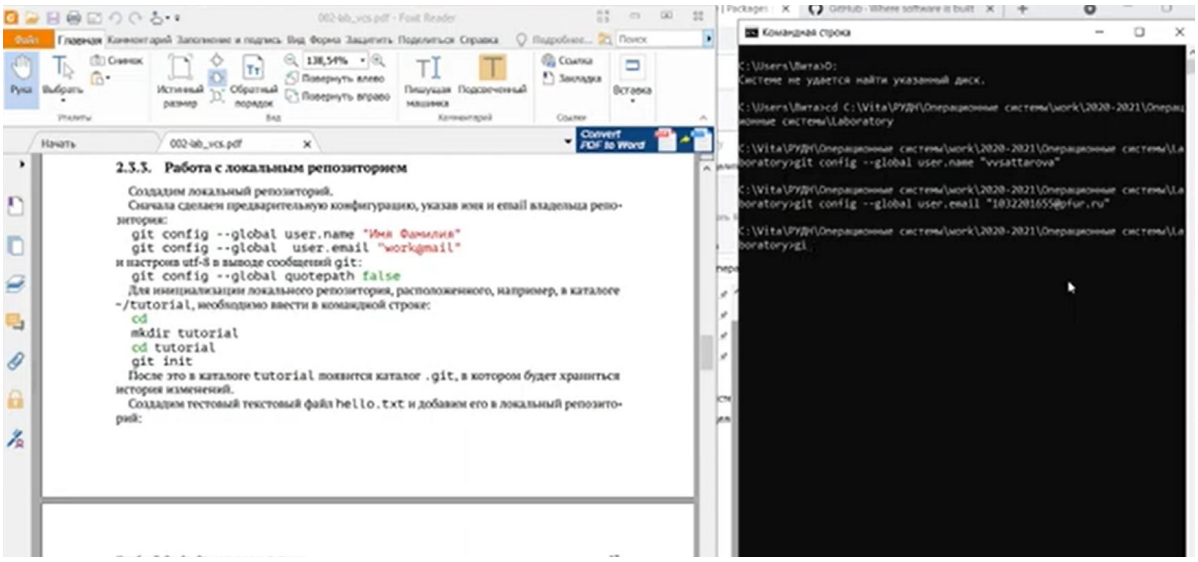

Figure 6.3: Рис. 3

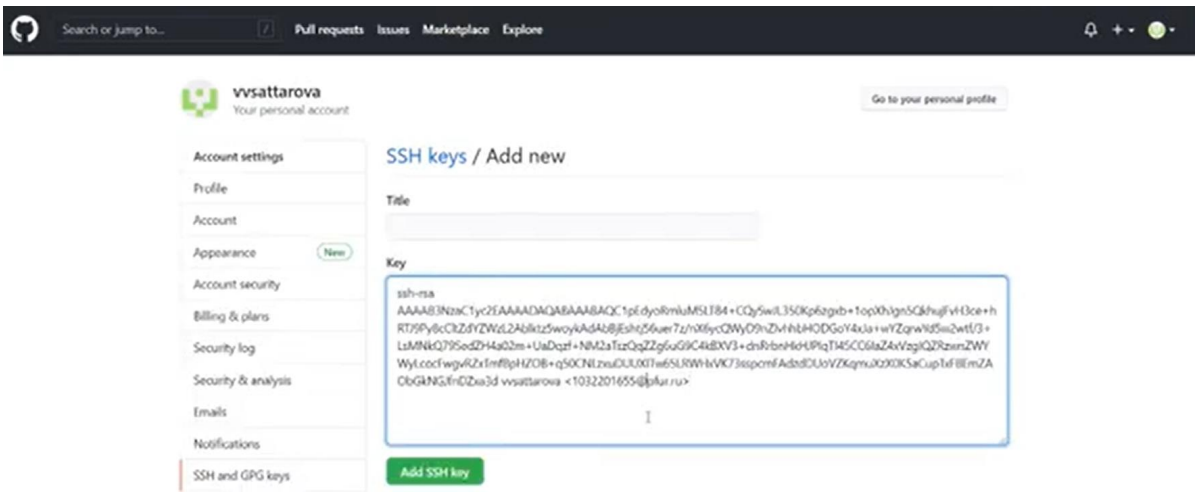

Figure 6.4: Рис. 4

4. Создала файл README.md, первый коммит, сделала первую публикацию на Гитхабе и настроила работу с ним через командную строку. (рис. -fig. 6.5)

Figure 6.5: Рис. 5

5. Добавила лицензию и игнорируемые файлы. (рис. -fig. 6.6)

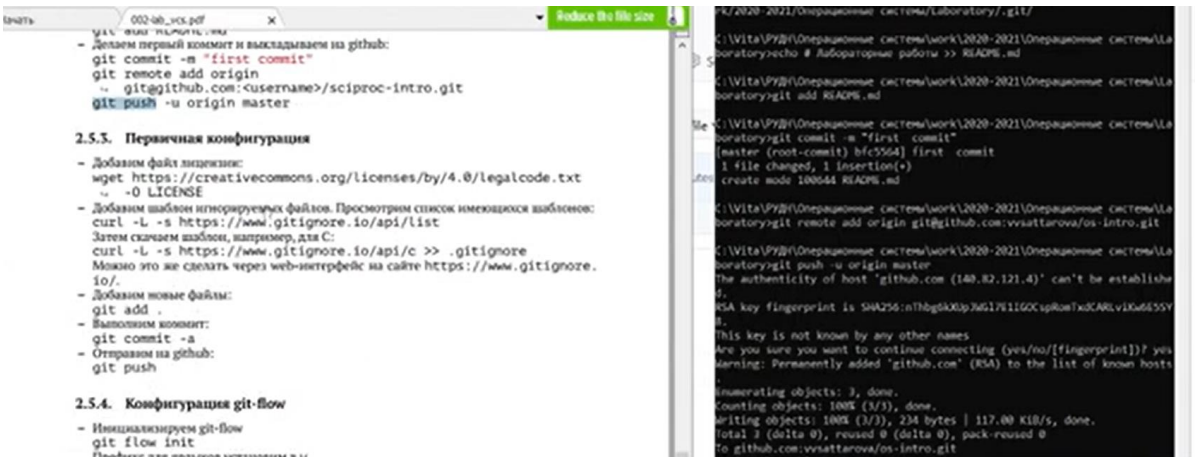

Figure 6.6: Рис. 6

6. Изучила git flow, создала две ветки master и develop, провела первый релиз и создала файл с версией. (рис. -fig. 6.7)

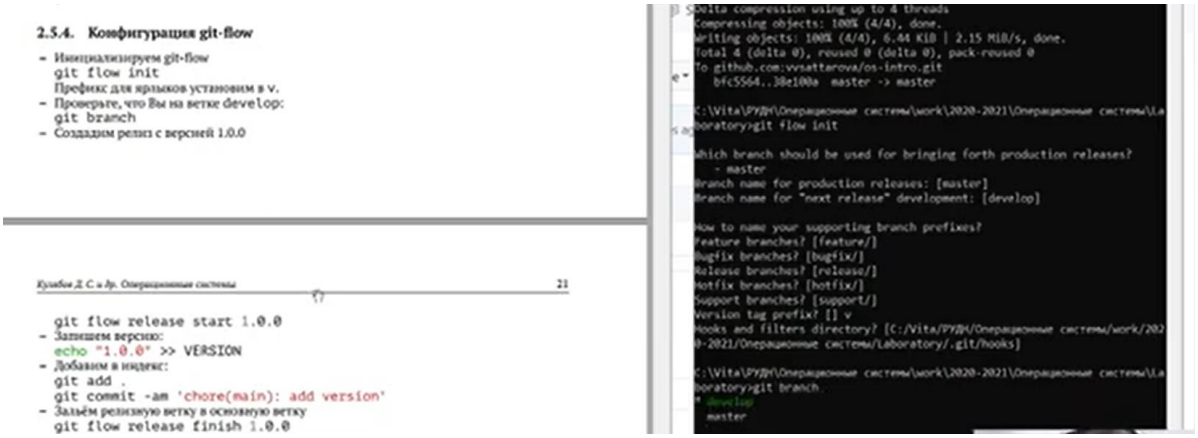

Figure 6.7: Рис. 7

7. Окончательно выпустила первый релиз при помощи тага. (рис. -fig. 6.8)

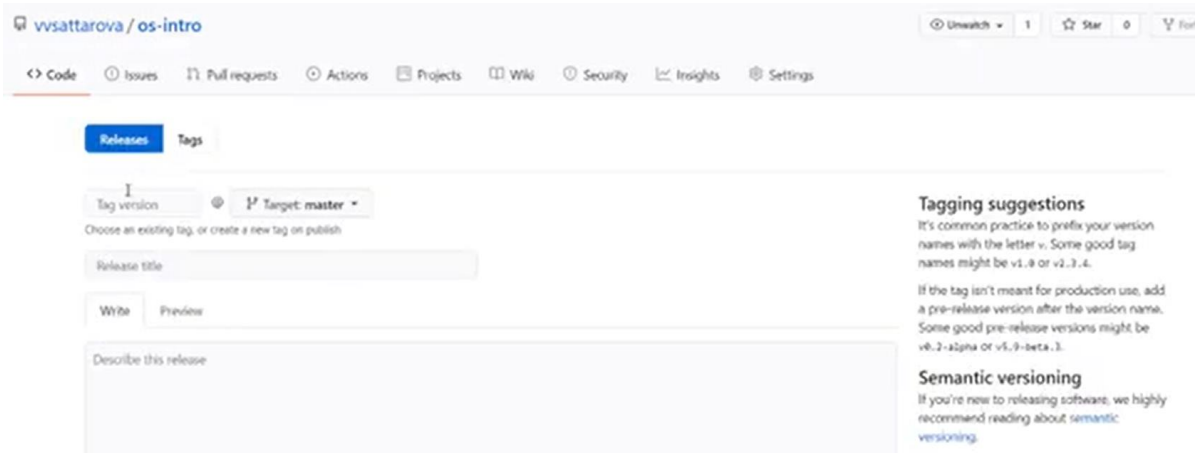

Figure 6.8: Рис. 8

8. Исправила ошибку, связанную с неправильной кодировкой и получила корректно отображающийся README. (рис. -fig. 6.9)

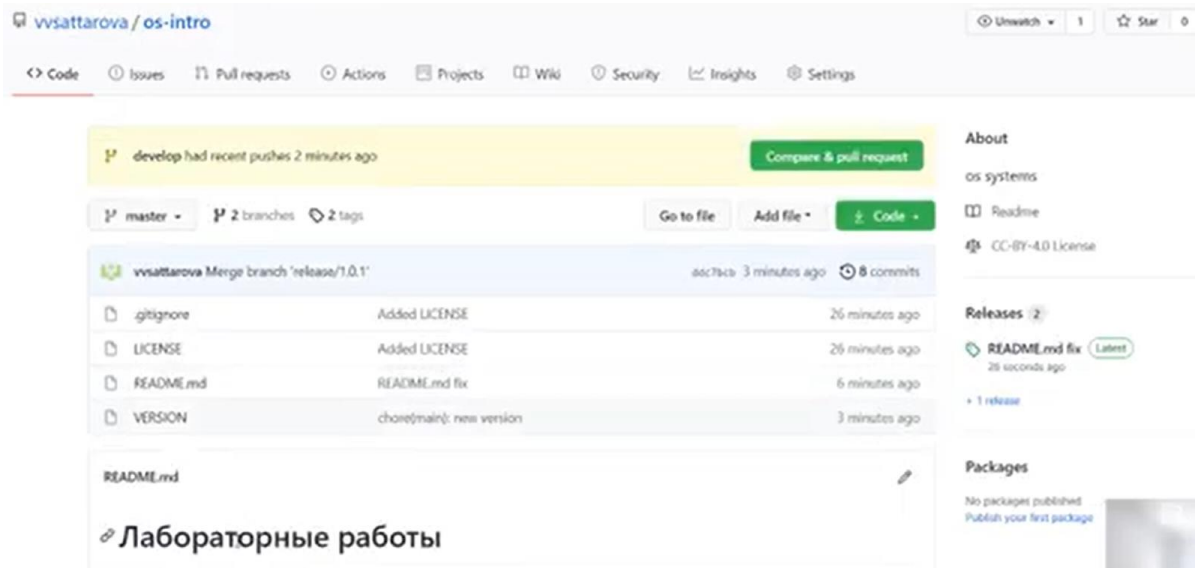

Figure 6.9: Рис. 9

7 Полученные результаты

Изучена информация, касающаяся контроля версий, настроен свой репозиторий на Гитхаб, создан первый релиз.

8 Анализ результатов

Работу получилось выполнить по инструкции, Гитхаб работает корректно, файлы отображаются правильно.

9 Заключение и выводы

В результате работы была изучена идеология и применение средств контроля версий, настроен репозиторий на Github, с которым возможно продолжать дальнейшую работу.