

Лабораторная работа No3.

Система контроля версий Git

Никифоров Георгий Сергеевич

Содержание

1 Цель работы 3

2 Задание 3

3 Теоретическое введение 3

4 Выполнение лабораторной работы 5

5 Вывод 8

Источники информации

3.1. Цель работы

Целью работы является изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой git.

3.2. Теоретическое введение

3.2.1. Системы контроля версий. Общие понятия

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. 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. 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.

3.2.2. Система контроля версий Git

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.

3.2.3. Стандартные процедуры работы при наличии центрального репозитория

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):

```
git checkout master  
git pull
```

```
git checkout -b имя_ветки
```

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 центральной репозитории. Для этого необходимо проверить, какие файлы изменились к текущему моменту:

```
git status
```

и при необходимости удаляем лишние файлы, которые не хотим отправлять в центральный репозиторий.

Затем полезно просмотреть текст изменений на предмет соответствия правилам ведения чистых коммитов:

```
git diff
```

Если какие-либо файлы не должны попасть в коммит, то помечаем только те файлы, изменения которых нужно сохранить. Для этого используем команды добавления и/или удаления с нужными опциями:

```
git add имена_файлов
```

```
git rm имена_файлов
```

Если нужно сохранить все изменения в текущем каталоге, то используем:

```
git add .
```

Затем сохраняем изменения, поясняя, что было сделано:

```
git commit -am "Some commit message"
```

и отправляем в центральный репозиторий:

```
git push origin имя_ветки
```

или

```
git push
```

Выполнение лабораторной работы:

1. Создал учетную запись github



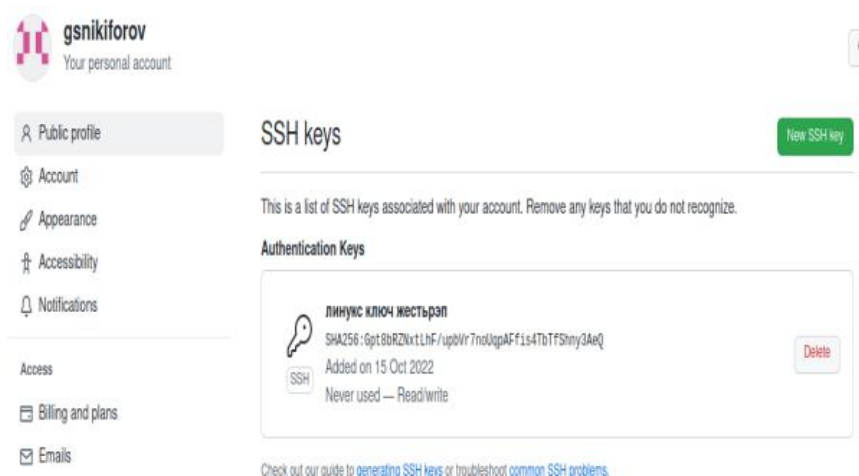
2. Настроил git

```
gsnikiforov@gsnikiforov-VirtualBox:~$ git config --global user.name "gsnikiforov"
gsnikiforov@gsnikiforov-VirtualBox:~$ git config --global user.email "goshaniker38@gmail.com"
gsnikiforov@gsnikiforov-VirtualBox:~$ git config --global core.quotepath false
gsnikiforov@gsnikiforov-VirtualBox:~$ git config --global init.defaultBranch master
gsnikiforov@gsnikiforov-VirtualBox:~$ git config --global core.autocrlf input
gsnikiforov@gsnikiforov-VirtualBox:~$ git config --global core.safecrlf warn
gsnikiforov@gsnikiforov-VirtualBox:~$ ssh-keygen -C "Георгий Никифоров goshaniker38@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/gsnikiforov/.ssh/id_rsa): /.ssh
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Saving key "/.ssh" failed: Permission denied
gsnikiforov@gsnikiforov-VirtualBox:~$ ssh-keygen -C "Георгий Никифоров goshaniker38@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/gsnikiforov/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gsnikiforov/.ssh/id_rsa
```

3. Создал ключ SSH

```
SHA256:Gpt8bR2NxtLhF/upbVr7noUpAFfIs4TbTfShny3AeQ Георгий Никифоров goshaniker38@gmail.com
The key's randomart image is:
+--[RSA 3072]-----+
  . .
  . + . O |
  O . * O |
  . E= . ..|
  . * S=... ..|
  . O=X.B.O.O.|
  O=O.O+O +.+|
  +..O. .+ +|
  . ... ++|
+-----[SHA256]-----+
gsnikiforov@gsnikiforov-VirtualBox:~$ ll ~/.ssh/
ls: невозможно получить доступ к '/home/gsnikiforov/.ssh/': Нет такого файла или каталога
gsnikiforov@gsnikiforov-VirtualBox:~$ ^C
gsnikiforov@gsnikiforov-VirtualBox:~$ ll ~/.ssh/
total 16
-rwxr-xr-x 2 gsnikiforov gsnikiforov 4096 окт 15 20:58 ./
-rwxr-xr-x 16 gsnikiforov gsnikiforov 4096 окт 15 20:55 ./
-rw-r--r-- 1 gsnikiforov gsnikiforov 2655 окт 15 20:58 id_rsa
-rw-r--r-- 1 gsnikiforov gsnikiforov 610 окт 15 20:58 id_rsa.pub
gsnikiforov@gsnikiforov-VirtualBox:~$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCA8c20ZTgsIw7iENNFUVRlV19jMWNqF0h54AHMOvAZQ+zkKhG1GZAIuV8orUXAk
h3TjFEMzRlHmIL5kOnRecBUphVzrwLUCXm4vg+kQK2isVlyPDPFwsXAFI2LDC18NNF+3ry2H0nZXGSTRsKXbd9u0/Q08BH5STT
cpQWrf380pRWLk8U5LI4Mh/UZ08l+k7Y/0gEifgKkWRf3100PDYh8BAdLauXgsgbyLkFhnSLffENz5qjGnczYlZ6WBDJ169kC6YH
gsnikiforov@gsnikiforov-VirtualBox:~$
```

4. Создал рабочее пространство и репозиторий курса на основе шаблона



gsnikiforov
Your personal account

Public profile

Account

Appearance

Accessibility

Notifications

Access

Billing and plans

Emails

SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

SSH key	Added	Used	Actions
линукс ключ жестърэл SHA256:Gpt8bR2NxtLhF/upbVr7noUpAFfIs4TbTfShny3AeQ	Added on 15 Oct 2022	Never used — Read/write	Delete

Check out our guide to [generating SSH keys](#) or [troubleshoot common SSH problems](#).

5. Создал рабочее пространство

```
gsnikiforov@gsnikiforov-VirtualBox:~$ mkdir -p ~/work/study/2022-2023/"Архитектура компьютера"
gsnikiforov@gsnikiforov-VirtualBox:~$ cd ~/work/study/2022-2023/"Архитектура компьютера"
```

6. Создал копию репозитория

```
gsnikiforov@gsnikiforov-VirtualBox:~$ git clone --recursive git@github.com:gsnikiforov/study_2022-2023_arh-pc.git
Клонирование в «study_2022-2023_arh-pc»...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 26 (delta 0), reused 17 (delta 0), pack-reused 0
Получение объектов: 100% (26/26), 16.39 Киб | 16.39 Миб/с, готово.
Подмодуль «template/presentation» (https://github.com/yanadharna/academic-presentation-markdown-template.git) заархивирован
Подмодуль «template/report» (https://github.com/yanadharna/academic-laboratory-report-template.git) заархивирован
Клонирование в «/home/gsnikiforov/study_2022-2023_arh-pc/template/presentation»...
remote: Enumerating objects: 71, done.
remote: Counting objects: 100% (71/71), done.
remote: Compressing objects: 100% (49/49), done.
remote: Total 71 (delta 23), reused 68 (delta 20), pack-reused 0
Получение объектов: 100% (71/71), 88.89 Киб | 1.53 Миб/с, готово.
Определение изменений: 100% (23/23), готово.
Клонирование в «/home/gsnikiforov/study_2022-2023_arh-pc/template/report»...
remote: Enumerating objects: 78, done.
remote: Counting objects: 100% (78/78), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 78 (delta 31), reused 69 (delta 22), pack-reused 0
Получение объектов: 100% (78/78), 292.27 Киб | 2.36 Миб/с, готово.
Определение изменений: 100% (31/31), готово.
Submodule path 'template/presentation': checked out '2703b47423792d472694aaf7555a5626dce51a25'
Submodule path 'template/report': checked out 'df7b2ef80f8def3b9a496f8695277469a1a7842a'
```

Самостоятельная работа.
Я НИЧЕГО НЕ ПОНЯЛ!!!!!! Рэп

Вывод: Я изучил идеологию и применение средств контроля версий. Приобрел практические навыки по работе с системой git