

Отчёт по лабораторной работе №2

Управление версиями

Корпаев Бегдурды

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Вывод	13
4	Контрольные вопросы	14

Список иллюстраций

2.1	Загрузка пакетов	6
2.2	Параметры репозитория	6
2.3	rsa-4096	7
2.4	ed25519	8
2.5	GPG ключ	8
2.6	GPG ключ	9
2.7	Параметры репозитория	10
2.8	Связь репозитория с аккаунтом	11
2.9	Загрузка шаблона	11
2.10	Первый коммит	12

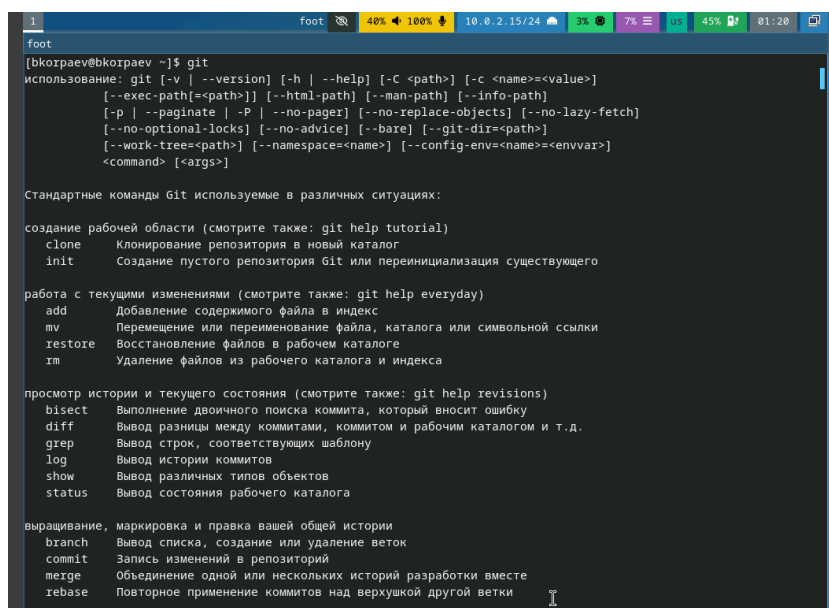
Список таблиц

1 Цель работы

Целью данной работы является изучение идеологии и применения средств контроля версий и освоение умений работать с git.

2 Выполнение лабораторной работы

Устанавливаем git, git-flow и gh.



```
foot
[bkorpaev@bkorpaev ~]$ git
использование: git [-v | --version] [-h | --help] [-C <path>] [--c <name>=<value>]
[--exec-path=<path>] [--html-path] [--man-path] [--info-path]
[-p | --paginate | -P | --no-pager] [--no-replace-objects] [--no-lazy-fetch]
[--no-optional-loops] [--no-advice] [--bare] [--git-dir=<path>]
[--work-tree=<path>] [--namespace=<name>] [--config-env=<name>=<envvar>]
<command> [<args>]

Стандартные команды Git используемые в различных ситуациях:

создание рабочей области (смотрите также: git help tutorial)
clone      Клонирование репозитория в новый каталог
init       Создание пустого репозитория Git или переинициализация существующего

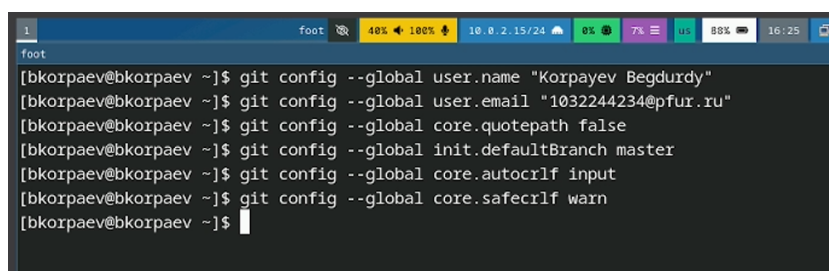
работа с текущими изменениями (смотрите также: git help everyday)
add        Добавление содержимого файла в индекс
mv         Перемещение или переименование файла, каталога или символической ссылки
restore    Восстановление файлов в рабочем каталоге
rm         Удаление файлов из рабочего каталога и индекса

просмотр истории и текущего состояния (смотрите также: git help revisions)
bisect     Выполнение двоичного поиска коммита, который вносит ошибку
diff       Вывод разницы между коммитами, коммитом и рабочим каталогом и т.д.
grep       Вывод строк, соответствующих шаблону
log        Вывод истории коммитов
show       Вывод различных типов объектов
status     Вывод состояния рабочего каталога

выращивание, маркировка и правка вашей общей истории
branch     Вывод списка, создание или удаление веток
commit     Запись изменений в репозиторий
merge      Объединение одной или нескольких историй разработки вместе
rebase     Повторное применение коммитов над вершиной другой ветки
```

Рис. 2.1: Загрузка пакетов

Зададим имя и email владельца репозитория, кодировку и прочие параметры.



```
foot
[bkorpaev@bkorpaev ~]$ git config --global user.name "Korpayev Begdurdy"
[bkorpaev@bkorpaev ~]$ git config --global user.email "1032244234@pfur.ru"
[bkorpaev@bkorpaev ~]$ git config --global core.quotePath false
[bkorpaev@bkorpaev ~]$ git config --global init.defaultBranch master
[bkorpaev@bkorpaev ~]$ git config --global core.autocrlf input
[bkorpaev@bkorpaev ~]$ git config --global core.safecrlf warn
[bkorpaev@bkorpaev ~]$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи

```
1 foot 40% 100% 10.0.2.15/24 0% 7% 86% 16:29
[bkorpaev@bkorpaev ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/bkorpaev/.ssh/id_rsa):
Enter passphrase for "/home/bkorpaev/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/bkorpaev/.ssh/id_rsa
Your public key has been saved in /home/bkorpaev/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Uc9v1NnTjo1SDR7W7j+gK8y+3IEjxmzNB15cld7I5L4 bkorpaev@bkorpaev
The key's randomart image is:
+---[RSA 4096]-----+
|      .  .  =  |
|      . o  =  Bo|
|      . B  Oo+ |
|      ...  o  *o|
|      .So  o  =.o|
|      + =  =  .|
|      Bo*  .. o  .|
|      o  ++o..E  ..|
|      .+oo.  .|
+---[SHA256]-----+
```

Рис. 2.3: rsa-4096

```

Enter passphrase for "/home/bkorpaev/.ssh/id_ed25519
" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/bkorpaev
/.ssh/id_ed25519
Your public key has been saved in /home/bkorpaev/.ss
h/id_ed25519.pub
The key fingerprint is:
SHA256:09E69nYyNLLBAfKzos9rT7L41y+r1EvPLM358BNaRoY b
korpaev@bkorpaev
The key's randomart image is:
+--[ED25519 256]--+
|
|      . .      |
|      o . .    |
|      o E.o    |
|      +S+.     |
|      . . .++*  |
|      . . .+BX o|
|      .o.*o*o*= .|
|      .+*++o=*==|
+-----[SHA256]-----+
[bkorpaev@bkorpaev ~]$ █

```

Рис. 2.4: ed25519

Создаем GPG ключ

```

[bkorpaev@bkorpaev ~]$ gpg --full-generate-key
gpg (GnuPG) 2.4.5; Copyright (C) 2024 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: создан каталог '/home/bkorpaev/.gnupg'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key from card
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096 █

```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

```
[bkorpaev@bkorpaev ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0
доверие: 0-, 0q, 0n, 0m, 0f, 1u
[keyboxd]
-----
sec   rsa4096/55A0F6B009F32938 2025-03-02 [SC]
      085911F10691881F4F1BB1FC55A0F6B009F32938
uid           [ абсолютно ] Korpayev Begdurdy <1032244234@pfur.ru>
ssb   rsa4096/596035F97E3FAA79 2025-03-02 [E]

[bkorpaev@bkorpaev ~]$ gpg --armor --export 085911F10691881F4F1BB1FC55A0F6B009F32938 | xclip -sel clip
[bkorpaev@bkorpaev ~]$ gpg --armor --export 085911F10691881F4F1BB1FC55A0F6B009F32938
```

Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```

NmFYzLZPUNh3
ZY/iNL7wm/MYtbuVkzCd+4iQXbfUJq20uXMcRMFbaPmeq0mvV6le
ox00qLkCysJr
zMaNAuL5RzTUZgUQZWORraEM4RtvsmZkCHgmVtpoMujLQXZNZNvj
J3YpylJ6fpxa
6xCRh8CRFbrUIQvn8dqviNACXadg5lYGWJE930uglTVlKljq659N
h60onLfsTZvk
7Y+IN8ju6Ase398FhtAsSzPcaAJt+0aCeKV33T97Y8jETVc2DrAi
8huslBGGKQtH
ESoQE3U4Ivu+Xkm5KPBBil6vIxcadvjwr/oolI3vB2sIbE3xmuRz
0U3E0+nvxh4a
ahAf7iV0RDAwS25CyYmtrReWQTa8MorAZQFBv4tUoSDV91487SB/
6FA055IcK6CP
REla5I6Y3q2+EzYrTE55scVZJR3VLXlGVAUHBf0Uiq5hWITCEDD
i4PXg8wZ+g9y
WP5VAQCqe/1DNTrwjitVuVMdp735X9cUV2vT5oWkiz02zJ+aXy/V
iD4e+enpZt3k
mqwZS4Am1i3H7ms0qj18BJPSoaWhbbf0bBumFMki5Swcr6HEUt/G
rUAi0IRo8AUS
tFSh1Do=
=aU9k
-----END PGP PUBLIC KEY BLOCK-----
[bkorpaev@bkorpaev ~]$

```

Рис. 2.7: Параметры репозитория

Настройка gh

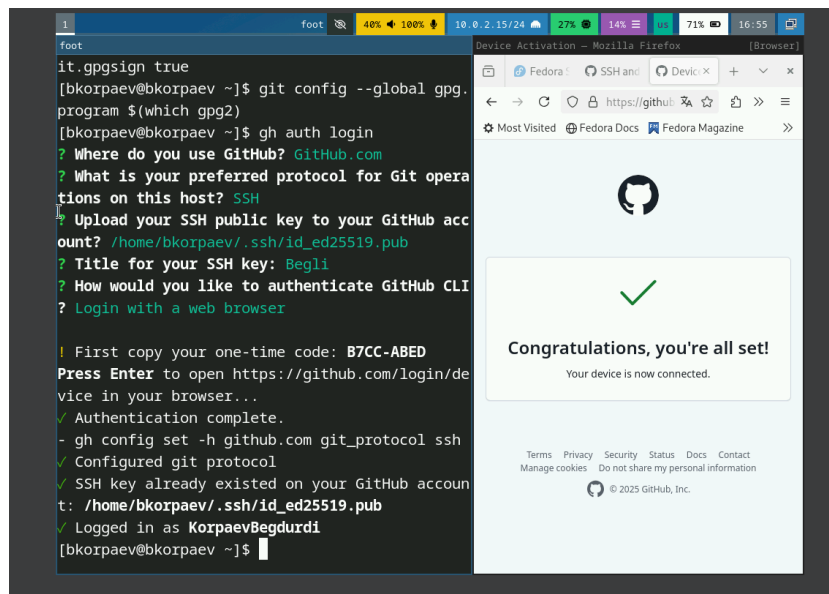


Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

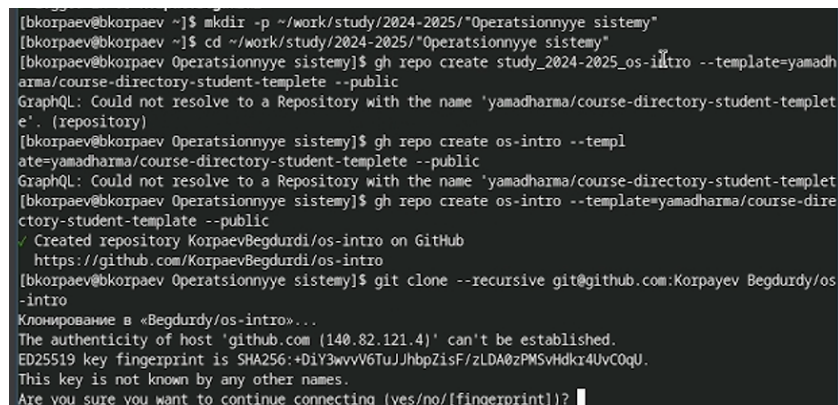


Рис. 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений

```
[bkorpaev@bkorpaev os_intro]$ git push
Перечисление объектов: 39, готово.
Подсчет объектов: 100% (39/39), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 342.31 КиБ | 2.44 МиБ/с, готово.
Total 38 (delta 4), reused 1 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:KorpaevBegdurdy/os-intro.git
   d6ced0a..20cdf77  master -> master
[bkorpaev@bkorpaev os_intro]$
```

Рис. 2.10: Первый коммит

3 Вывод

Мы приобрели практические навыки работы с сервисом github.

4 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?

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

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

- хранилище - пространство на накопителе где расположен репозиторий
- commit - сохранение состояния хранилища
- история - список изменений хранилища (коммитов)
- рабочая копия - локальная копия сетевого репозитория, в которой работает программист. Текущее состояние файлов проекта, основанное на версии, загруженной из хранилища (обычно на последней)

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

Централизованные системы контроля версий представляют собой приложения типа клиент-сервер, когда репозиторий проекта существует в единственном экземпляре и хранится на сервере. Доступ к нему осуществлялся через специальное клиентское приложение. В качестве примеров таких программных продуктов можно привести CVS, Subversion.

Распределенные системы контроля версий (Distributed Version Control System, DVCS) позволяют хранить репозиторий (его копию) у каждого разработчика, работающего с данной системой. При этом можно выделить центральный репозиторий (условно), в который будут отправляться изменения из локальных и, с ним же эти локальные репозитории будут синхронизироваться. При работе с такой системой, пользователи периодически синхронизируют свои локальные репозитории с центральным и работают непосредственно со своей локальной копией. После внесения достаточного количества изменений в локальную копию они (изменения) отправляются на сервер. При этом сервер, чаще всего, выбирается условно, т.к. в большинстве DVCS нет такого понятия как “выделенный сервер с центральным репозиторием”.

4. Опишите действия с VCS при единоличной работе с хранилищем.

Один пользователь работает над проектом и по мере необходимости делает коммиты, сохраняя определенные этапы.

5. Опишите порядок работы с общим хранилищем VCS.

Несколько пользователей работают каждый над своей частью проекта. При этом каждый должен работать в своей ветки. При завершении работы ветка пользователя сливается с основной веткой проекта.

6. Каковы основные задачи, решаемые инструментальным средством git?

- Ведение истории версий проекта: журнал (log), метки (tags), ветвления (branches).

- Работа с изменениями: выявление (diff), слияние (patch, merge).
- Обеспечение совместной работы: получение версии с сервера, загрузка обновлений на сервер.

7. Назовите и дайте краткую характеристику командам git.

- git config - установка параметров
- git status - полный список изменений файлов, ожидающих коммита
- git add . - сделать все измененные файлы готовыми для коммита.
- git commit -m "[descriptive message]" - записать изменения с заданным сообщением.
- git branch - список всех локальных веток в текущей директории.
- git checkout [branch-name] - переключиться на указанную ветку и обновить рабочую директорию.
- git merge [branch] — соединить изменения в текущей ветке с изменениями из заданной.
- git push - запустить текущую ветку в удаленную ветку.
- git pull - загрузить историю и изменения удаленной ветки и произвести слияние с текущей веткой.

8. Приведите примеры использования при работе с локальным и удалённым репозиториями.

- git remote add [имя] [url] — добавляет удалённый репозиторий с заданным именем;
- git remote remove [имя] — удаляет удалённый репозиторий с заданным именем;
- git remote rename [старое имя] [новое имя] — переименовывает удалённый репозиторий;
- git remote set-url [имя] [url] — присваивает репозиторию с именем новый адрес;

- `git remote show [имя]` — показывает информацию о репозитории.

9. Что такое и зачем могут быть нужны ветви (branches)?

Ветвление — это возможность работать над разными версиями проекта: вместо одного списка с упорядоченными коммитами история будет расходиться в определённых точках. Каждая ветвь содержит легковесный указатель HEAD на последний коммит, что позволяет без лишних затрат создать много веток. Ветка по умолчанию называется `master`, но лучше назвать её в соответствии с разрабатываемой в ней функциональностью.

10. Как и зачем можно игнорировать некоторые файлы при `commit`?

Зачастую нам не нужно, чтобы Git отслеживал все файлы в репозитории, потому что в их число могут входить: