

Лабораторная работы №2.

Markdown

ДИОН ГОНССАН СЕДРИК МИШЕЛ

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Вывод	11
4	Контрольные вопросы	12
	Список литературы	16

Список иллюстраций

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

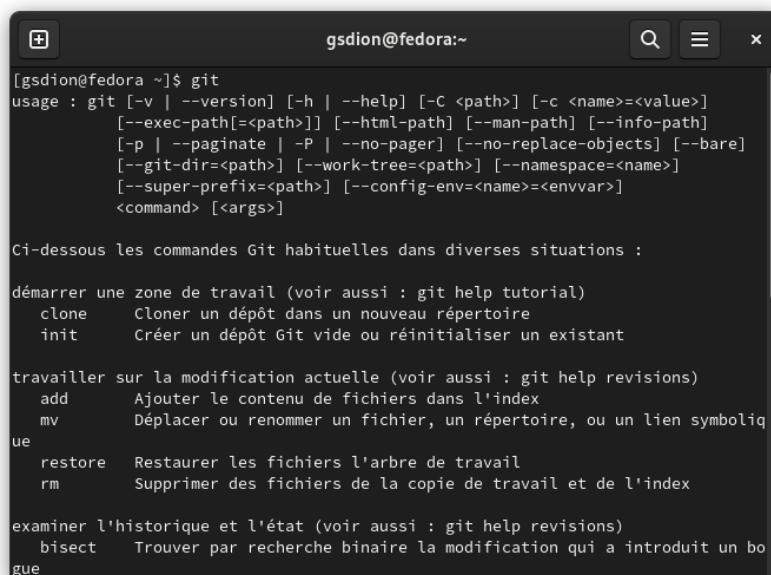
Список таблиц

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.



```
[gsdion@fedora ~]$ git
usage : 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] [--bare]
          [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
          [--super-prefix=<path>] [--config-env=<name>=<envvar>]
          <command> [<args>]

Ci-dessous les commandes Git habituelles dans diverses situations :

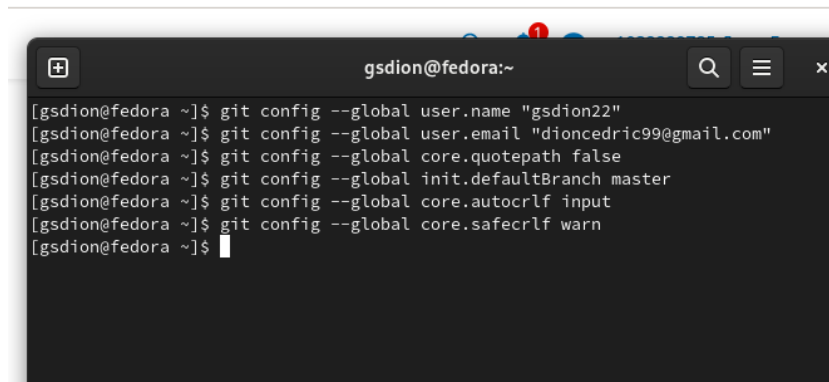
démarrer une zone de travail (voir aussi : git help tutorial)
  clone   Cloner un dépôt dans un nouveau répertoire
  init    Créer un dépôt Git vide ou réinitialiser un existant

travailler sur la modification actuelle (voir aussi : git help revisions)
  add     Ajouter le contenu de fichiers dans l'index
  mv      Déplacer ou renommer un fichier, un répertoire, ou un lien symbolique
  rm      Supprimer des fichiers de la copie de travail et de l'index

examiner l'historique et l'état (voir aussi : git help revisions)
  bisect  Trouver par recherche binaire la modification qui a introduit un bogue
```

Рис. 2.1: Загрузка пакетов

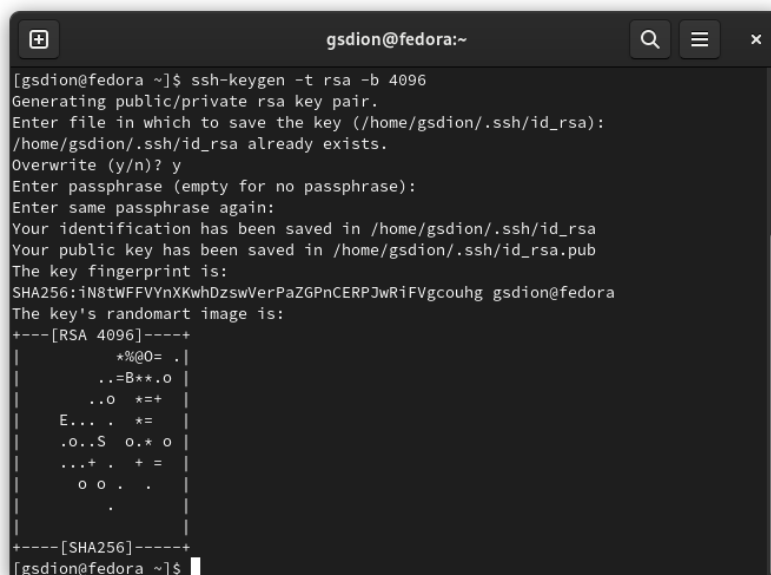
Зададим имя и email владельца репозитория, кодировку и прочие параметры.

A terminal window titled 'gsdion@fedora:~' with search, menu, and close icons. It contains a series of git configuration commands and their outputs.

```
[gsdion@fedora ~]$ git config --global user.name "gsdion22"
[gsdion@fedora ~]$ git config --global user.email "dioncedric99@gmail.com"
[gsdion@fedora ~]$ git config --global core.quotepath false
[gsdion@fedora ~]$ git config --global init.defaultBranch master
[gsdion@fedora ~]$ git config --global core.autocrlf input
[gsdion@fedora ~]$ git config --global core.safecrlf warn
[gsdion@fedora ~]$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи

A terminal window titled 'gsdion@fedora:~' with search, menu, and close icons. It shows the execution of the ssh-keygen command to generate an RSA key pair, including prompts for file location, passphrase, and the resulting key fingerprint and randomart image.

```
[gsdion@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/gsdion/.ssh/id_rsa):
/home/gsdion/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gsdion/.ssh/id_rsa
Your public key has been saved in /home/gsdion/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:iN8tWFFVYnXKwhDzswVerPaZGPnCERPJwRiFVgcouhg gsdion@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|          *%@0= . |
|         ..=B*+.o |
|        ..O  *+= |
|       E... .  *= |
|      .o..S  o.* o |
|     ...+ .  + = |
|      o o .  . |
|      . |
+-----[SHA256]-----+
[gsdion@fedora ~]$
```

Рис. 2.3: rsa-4096

```
gsdion@fedora:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/gsdion/.ssh/id_ed25519):
/home/gsdion/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gsdion/.ssh/id_ed25519
Your public key has been saved in /home/gsdion/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:GDJjVdah2siZdKOA9kMcPxLAtRb6SwNlGI83WFlr1E gsdion@fedora
The key's randomart image is:
+--[ED25519 256]--+
|  oo+oo+E         |
|  *.***+=  .      |
|  .+o0= = + .     |
|  ++*% + o        |
|  +$.* o          |
|  ..* o           |
|                  |
+-----[SHA256]-----+
[gsdion@fedora ~]$
```

Рис. 2.4: ed25519

Создаем GPG ключ

```
gsdion@fedora:~$ gpg --full-gen-key
Commentaire :
Vous avez sélectionné cette identité :
« gsdion22 <dioncedric99@gmail.com> »

Changer le (N)om, le (C)ommentaire, l'(A)dresse électronique
ou (O)ui/(Q)uitter ? o
De nombreux octets aléatoires doivent être générés. Vous devriez faire
autre chose (taper au clavier, déplacer la souris, utiliser les disques)
pendant la génération de nombres premiers ; cela donne au générateur de
nombres aléatoires une meilleure chance d'obtenir suffisamment d'entropie.
De nombreux octets aléatoires doivent être générés. Vous devriez faire
autre chose (taper au clavier, déplacer la souris, utiliser les disques)
pendant la génération de nombres premiers ; cela donne au générateur de
nombres aléatoires une meilleure chance d'obtenir suffisamment d'entropie.
gpg: revocation certificate stored as '/home/gsdion/.gnupg/openpgp-revocs.d/8FE7
F89FD00123159EFCB2390448F62B926A0A59.rev'
les clefs publique et secrète ont été créées et signées.

pub   rsa4096 2023-02-25 [SC]
      8FE7F89FD00123159EFCB2390448F62B926A0A59
uid           gsdion22 <dioncedric99@gmail.com>
sub     rsa4096 2023-02-25 [E]

[gsdion@fedora ~]$
```

Рис. 2.5: GPG ключ

Добавляем GPG ключ в аккаунт

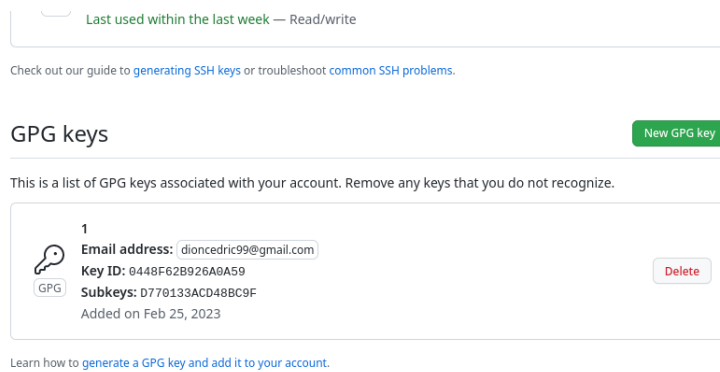


Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```
gsdion@fedora ~]$ gpg --list-secret-keys --keyid-format LONG
/home/gsdion/.gnupg/pubring.kbx
-----
sec   rsa4096/2FC56BAAFD3A5D0F 2023-02-24 [SC]
      701609292B76111CE9AB223F2FC56BAAFD3A5D0F
uid           [ ultimate ] gsdion22 <dioncedric99@gmail.com>
ssb   rsa4096/F766E594ABB90BE2 2023-02-24 [E]

sec   rsa4096/0448F62B926A0A59 2023-02-25 [SC]
      8FE7F89FD00123159EFCB2390448F62B926A0A59
uid           [ ultimate ] gsdion22 <dioncedric99@gmail.com>
ssb   rsa4096/D770133ACD48BC9F 2023-02-25 [E]

[gsdion@fedora ~]$ git config --global user.signingkey 0448F62B926A0A59
[gsdion@fedora ~]$ git config --global commit.gpgsign true
[gsdion@fedora ~]$ git config --global gpg.program $(which gpg2)
[gsdion@fedora ~]$
```

Рис. 2.7: Параметры репозитория

Настройка gh

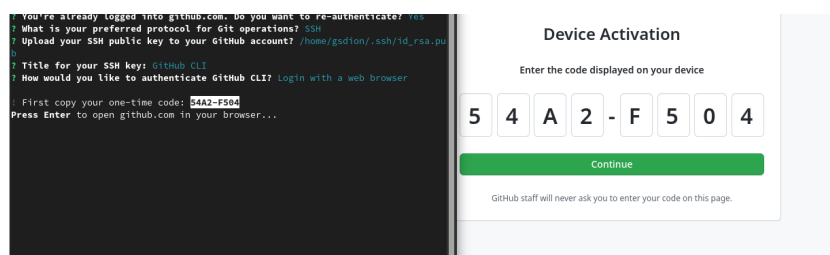


Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

```
gsdion@fedora:~/work/study/2022-2023/Операционные сист...
Sous-module 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git) enregistré pour le chemin 'template/report'
Clonage dans '/home/gsdion/work/study/2022-2023/Операционные системы/os-intro/template/presentation'...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Réception d'objets: 100% (82/82), 92.90 Kio | 1011.00 Kio/s, fait.
Résolution des deltas: 100% (28/28), fait.
Clonage dans '/home/gsdion/work/study/2022-2023/Операционные системы/os-intro/template/report'...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Réception d'objets: 100% (101/101), 327.25 Kio | 2.29 Mio/s, fait.
Résolution des deltas: 100% (40/40), fait.
Chemin de sous-module 'template/presentation' : 'b1be3800ee91f5809264cb755d316174540b753e' extrait
Chemin de sous-module 'template/report' : '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2' extrait
[gsdion@fedora Операционные системы]$
```

Рис. 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений

```
py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
[gsdion@fedora os-intro]$ git push
Énumération des objets: 40, fait.
Décompte des objets: 100% (40/40), fait.
Compression des objets: 100% (30/30), fait.
Écriture des objets: 100% (38/38), 343.04 Kio | 3.12 Mio/s, fait.
Total 38 (delta 4), réutilisés 0 (delta 0), réutilisés du pack 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:gsdion22/study_2022-2023_os-intro.git
4775910..bf64509 master -> master
[gsdion@fedora os-intro]$
```

- Создайте необходимые каталоги:

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:

Список литературы

1. Лекция Системы контроля версий
2. GitHub для начинающих