Лабораторная работа №2 Первоначальна настройка git

Дисциплина Операционные системы

Агарков Артём Сергеевич НПМбв-02-20"

Содержание"

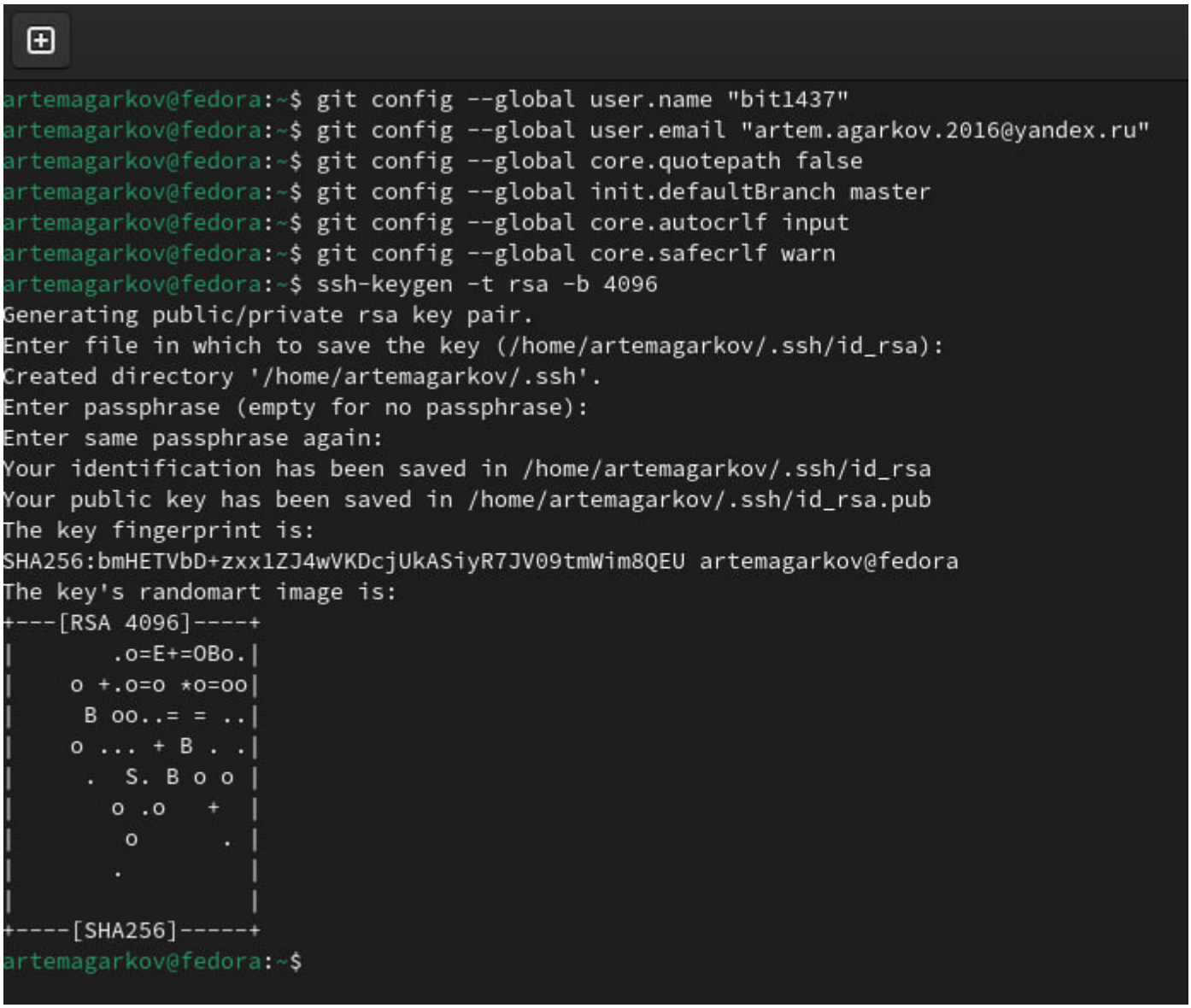
Цель работы

- Изучить применение средств контроля версий
- Освоить умения по работе с git'ом

Выполнение лабораторной работы

Настройка GIT

Сконфигурируем git и создадим SSH ключ



Создадим PGP ключ

```
   (3) DSA (sign only)
   (4) RSA (sign only)
   (9) ECC (sign and encrypt) *default*
  (10) ECC (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N)
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: Artem Agarkoff
Email address: artem.agarkov.2016@yandex.ru
Comment:
You selected this USER-ID:
    "Artem Agarkoff <artem.agarkov.2016@yandex.ru>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

gpg: /home/artemagarkov/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/home/artemagarkov/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/artemagarkov/.gnupg/openpgp-revocs.d/F5E18
public and secret key created and signed.

pub   rsa4096 2024-05-20 [SC]
      F5E186B63838F0382CA6FD7CD310E8A8137C1A3E
uid                      Artem Agarkoff <artem.agarkov.2016@yandex.ru>
sub   rsa4096 2024-05-20 [E]

artemagarkov@fedora:~$
artemagarkov@fedora:~$ S
```

С помощью команды gpg --armor --export | xclip -sel clip скопируем PGP ключ. И вставим в наш Github.

Bit1437 (Bit1437)
Your personal account

- Public profile
- Account
- Appearance
- Accessibility
- Notifications

Access

- Billing and plans
- Emails
- Password and authentication
- Sessions
- SSH and GPG keys
- Organizations
- Enterprises
- Moderation

Code, planning, and automation

- Repositories
- Codespaces
- Packages
- Copilot
- Pages
- Saved replies

Security

- Code security and analysis

Integrations

- Applications
- Scheduled reminders

Archives

- Security log
- Sponsorship log

- Developer settings

SSH keys

This is a list of SSH keys associated with your account. Remove any k

Authentication keys

MySSH
SHA256:VNi1FNrK7IuoeHn6krLck6Ptj6N0EgrL0DbQ5351RV0
SSH
Added on Mar 24, 2024
Last used within the last 2 months — Read/write

Check out our guide to connecting to GitHub using SSH keys or troubleshoot co

GPG keys

This is a list of GPG keys associated with your account. Remove any k

Email address: artem.agarkov.2016@yandex.ru
Key ID: 03D86406F3B3FB67
Subkeys: 836DF709F2D5677E
GPG
Added on May 20, 2024

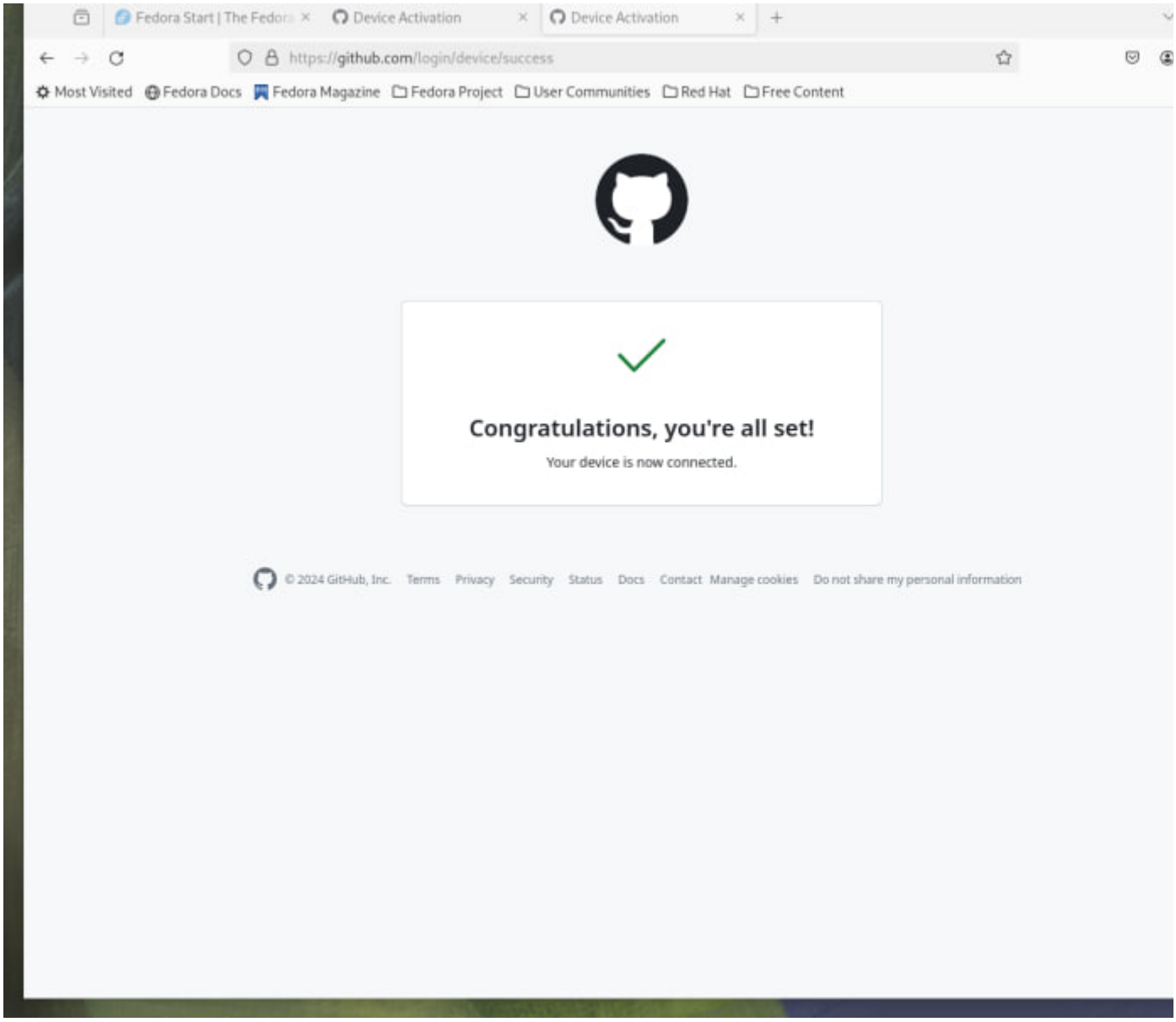
MyGPG
Email address: artem.agarkov.2016@yandex.ru
Key ID: E4B6C71D3188D6DD
Subkeys: 72935544276EC555
GPG
Added on Mar 24, 2024

Learn how to generate a GPG key and add it to your account.

Vigilant mode

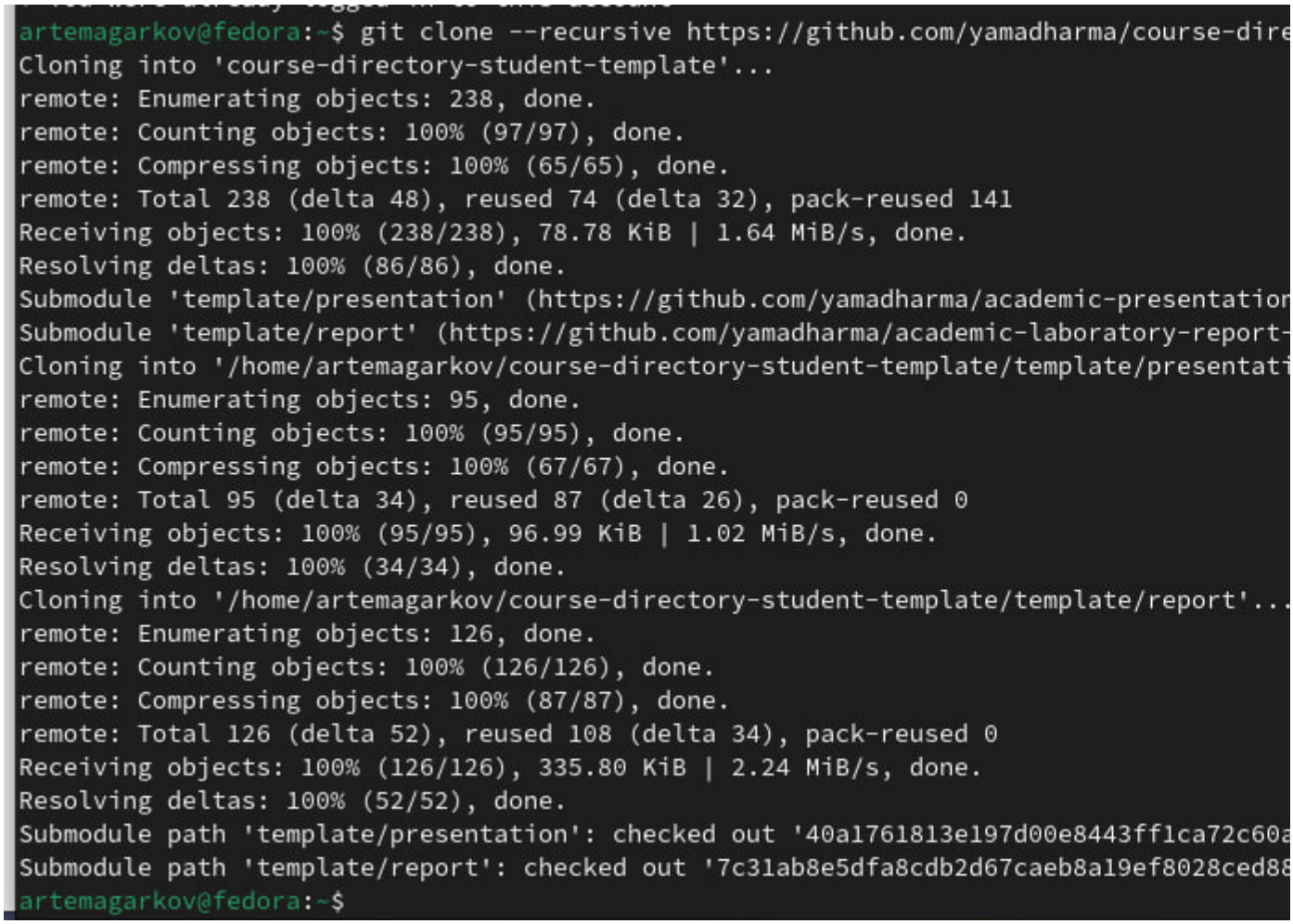
Flag unsigned commits as unverified
This will include any commit attributed to your account but not signed with
Note that this will include your existing unsigned commits.
Learn about vigilant mode.

Авторизируемся в Github с помощью комманды gh



Создание рабочего пространства

Склонируем репозиторий на локальную машину



Ответы на контрольные вопросы

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

Системы контроля версий (VCS) предназначены для отслеживания изменений в программном коде и обеспечения коллективной разработки.

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

Хранилище: Место, где хранятся все изменения и версии программного кода. Commit: Отдельное изменение или набор изменений в коде, зафиксированное в системе контроля версий. История: Последовательность коммитов, отображающая эволюцию кода. Рабочая копия: Локальная копия проекта, с которой работает разработчик.

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

Децентрализованные VCS копируют всю историю изменений на каждый клиентский компьютер, в то время как централизованные VCS хранят все изменения на центральном сервере и клиенты получают только последние версии файлов. Примеры децентрализованных VCS: Git, Mercurial. Примеры централизованных VCS: Subversion, CVS.

Опишите действия с VCS при единоличной работе с хранилищем.

При индивидуальной разработке пользователь клонирует проект на свой компьютер, вносит изменения и создает новые версии, коммитя их в системе контроля версий.

Опишите порядок работы с общим хранилищем VCS.

Пользователь получает версию проекта из центрального хранилища, вносит изменения, коммитит их и отправляет обратно в хранилище.

Каковы основные задачи, решаемые инструментальным средством git?

Git используется для разработки проектов в команде, контроля изменений в файлах и возможности сохранения нескольких состояний проекта.

Назовите и дайте краткую характеристику командам git.

git add - добавляет изменения для коммита.

git commit - сохраняет изменения в репозитории с названием.

git push - отправляет изменения на удаленный репозиторий.

git config - позволяет изменить настройки Git.

Приведите примеры использования при работе с локальным и удалённым репозиториями.

В локальном репозитории разработчик может вносить изменения в код и коммитить их без доступа к сети. В удаленном репозитории команда разработчиков может совместно работать над проектом, обмениваясь изменениями через централизованный сервер.

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

Ветви используются для параллельной разработки функций или исправлений, чтобы избежать конфликтов между изменениями и обеспечить безопасное тестирование нового кода.

Как и зачем можно игнорировать некоторые файлы при commit?

Файлы могут быть проигнорированы с помощью файла .gitignore, чтобы избежать загрязнения репозитория лишними или конфиденциальными файлами.

Выводы

Мы изучили идеологию применения средств контроля версий и освоили базовые комманды git'a.