

Операционные системы

Отчет к лабораторной работе №2

Управление версиями

Студент: Зевде Эйоб Аманте

Группа: НПИбд01-21

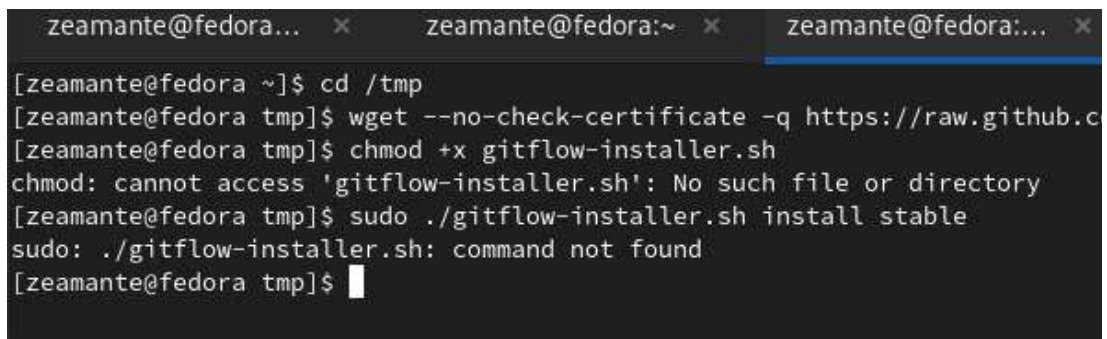
Цели работы

Изучить идеологию и применение средств контроля

версий, а также освоить умения по работе с git.

Выполнение работы

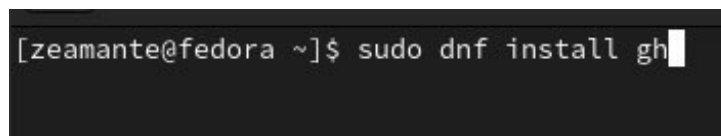
- Установка git-flow



```
zeamante@fedora... x  zeamante@fedora:~ x  zeamante@fedora:... x
[zeamante@fedora ~]$ cd /tmp
[zeamante@fedora tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/
[zeamante@fedora tmp]$ chmod +x gitflow-installer.sh
chmod: cannot access 'gitflow-installer.sh': No such file or directory
[zeamante@fedora tmp]$ sudo ./gitflow-installer.sh install stable
sudo: ./gitflow-installer.sh: command not found
[zeamante@fedora tmp]$
```

Рис 1. Установка в ручную git-flow

- Установка gh



```
[zeamante@fedora ~]$ sudo dnf install gh
```

Рис 2. Устанавливаем gh

- Базовая настройка git

```
git config --global user.name "Name Surname"
```

```
git config --global user.email "work@mail"
```

```
git config --global core.quotepath false
```

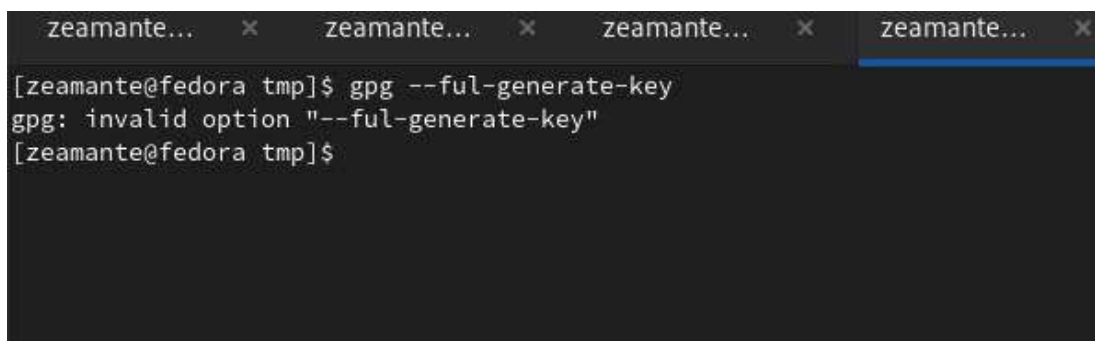
```
git config --global init.defaultBranch master
```

```
git config --global core.autocrlf input
```

```
git config --global core.safecrlf warn
```

С помощью данных команд зададим имя и email, настроим utf8, зададим имя начальной ветки, а также параметры autocrlf и safecrlf.

- Создаем ключи ssh с помощью команд `ssh-keygen -t rsa -b 4096` и `ssh-keygen -t ed25519`
- Создаем ключи pgp



```
zeamante... x zeamante... x zeamante... x zeamante... x
[zeamante@fedora tmp]$ gpg --ful-generate-key
gpg: invalid option "--ful-generate-key"
[zeamante@fedora tmp]$
```

Рис 3. Вводим программу и выбираем
интересующие нас значения

6. Добавление PGP ключа в GitHub

Рис 4. Копируем отпечаток ключа

Рис 5. Копируем наш ключ и вставляем в GitHub

7. Настраиваем автоматическую подпись коммитов

Рис 6. Используя введённый email, указываем Git
применять его при подписи коммитов

8. С помощью команды `gh auth login` авторизуемся и создаем репозиторий курса на основе шаблона

Рис 7. Создание репозитория примет следующий вид

9. Настраиваем каталог курса

Рис 8. С помощью данных команд настраиваем каталог

И с помощью данных команд

`git add .`

`git commit -am 'feat(main): make course structure'`

`git push`

отправляем файлы на сервер.

Выводы

В ходе выполнения данной лабораторной работы изучили идеологию и применение средств контроля версий, а также освоили умения по работе с git.

Контрольные вопросы

1. VCS – это программное обеспечение для облегчения работы с изменяющейся информацией, применяется при работе группы людей над одним проектом.

2. Хранилище – это место хранения всех версий и служебной информации

Commit – это процесс создания новой версии

История – это информация по изменениям, внесенным в проект

Рабочая копия – это текущее состояние файлов, основанное на версии, загруженной из хранилища.

3. Централизованные были созданы для бэкапирования, отслеживания и синхронизации файлов, а также все изменения в ней проходят через центральный сервер. Примеры: CVS, Subversion

Децентрализованные были созданы для обмена изменениями, и у каждого есть свой полноценный репозиторий. И у децентрализованных нет такой жестко заданной структуры репозитория с центральным сервером. Примеры: Git, Bazaar, Mercurial.

6. Есть 2 основные задачи: хранение информации о всех изменениях в коде и обеспечение удобства командной работы

7.

- создание основного дерева репозитория:

```
1 git init
```

- получение обновлений (изменений) текущего дерева из центрального репозитория:

```
1 git pull
```

- отправка всех произведённых изменений локального дерева в центральный репозиторий:

```
1 git push
```

- просмотр списка изменённых файлов в текущей директории:

```
1 git status
```

- просмотр текущих изменений:

```
1 git diff
```

- сохранение текущих изменений:

- добавить все изменённые и/или созданные файлы и/или каталоги:

```
1 git add .
```

- добавить конкретные изменённые и/или созданные файлы и/или каталоги:

```
1 git add имена_файлов
```

- удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории):

```
1 git rm имена_файлов
```

- сохранение добавленных изменений:

- сохранить все добавленные изменения и все изменённые файлы:

```
1 git commit -am 'Описание коммита'
```

8. Если вы захотите продолжить работу над проектом на другом устройстве то вам понадобится удаленный репозиторий.

9. При работе нескольких человек над одним проектом, ветви будут нужны чтобы они могли

спокойно работать, не мешая при этом друг другу.

10. Игнорируемые файлы – это обычно артефакты сборки или файлы, генерируемые машиной из исходных файлов в репозитории или же файлы которые не должны попасть в коммиты.