Лабораторная работа №3

Гэинэ Андрей

Содержание

Цель работы

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

Задание

Создать базовую конфигурацию для работы с git. - Создать ключ SSH. - Создать ключ PGP. - Настроить подписи git. - Зарегистрироваться на Github. - Создать локальный каталог для выполнения заданий по предмету

Порядок работы с общим хранилищем VCS:

- Клонировать репозиторий себе в гитхаб
- Клонировать репозиторий себе на устройство
- Внести изменения
- Добавить новую версию файлов на сервер

Git решает две задачи: хранить информацию о всех изменениях в коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

Краткая характеристика команд git:

- git config - настройки
- git init - создание репозитория
- git add - добавление файлов в индекс
- git commit - коммит изменений
- git status - список измененных файлов
- git push - перенос изменений в главную ветку
- git rm - удаление файлов из индекса

Локальный репозиторий можно загрузить на гитхаб и работать с ним с помощью VCS, т.е. загружать новые версии, не теряя старые.

Ветка в Git это подвижный указатель на один из коммитов. Обычно ветка указывает на последний коммит в цепочке коммитов. Ветки нужны для того, чтобы программисты могли вести совместную работу над проектом и не мешать друг другу при этом.

Чтобы проигнорировать файлы при коммит, надо просто не добавлять их в коммит. Игнорируют те файлы, которые пользователь не хочет отправлять в репозиторий.

Выполнение лабораторной работы

Скачиваем и устанавливаем git flow и gh.

```
[andre@andre ~]$ cd /tmp
[andre@andre tmp]$ wget --no-check-certificate -q https://raw.github.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[andre@andre tmp]$ chmod +x gitflow-installer.sh
[andre@andre tmp]$ sudo ./gitflow-installer.sh install stable
```

Рис. 1: Рис.1

Настраиваем git: задаем имя владельца, настраиваем utf-8 в выводе сообщений git, зададим имя начальной ветки, параметры autpcrlf и safecrlf.

```
[andre@andre tmp]$ git config --global user.name "Andrei Gaina"
[andre@andre tmp]$ git config --global user.email "andreigaina220@gmail.com"
[andre@andre tmp]$ git config --global core.quotepath false
[andre@andre tmp]$ git config --global init.defaultBranch master
[andre@andre tmp]$ git config --global core.autocrlf input
[andre@andre tmp]$ git config --global core.safecrlf warn
```

Рис. 2: Рис.2

Создаем ключ SSH с помощью команды ssh-keygen

```
[andre@andre tmp]$ ssh-keygen -t rsa -b 4096
```

```
[andre@andre tmp]$ ssh-keygen -t ed2551
```

Генерируем pgp ключ и вводим свои данные.

```
andre@andre tmp]$ gpg --armor --export C0FED5B2CEFA50BC | xclip -sel clip
```

Рис. 3: Рис.5

Добавляем ключ в github.

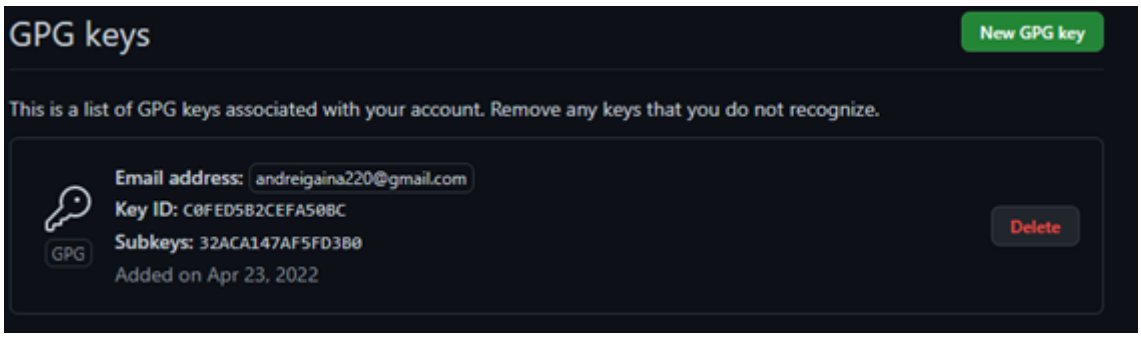

Рис. 4: Рис.6

Настраиваем подписи git.

```
[andre@andre tmp]$ git config --global user.signingkey C0FED5B2CEFA50BC
[andre@andre tmp]$ git config --global commit.gpgsign true
[andre@andre tmp]$ git config --global gpg.program $(which gpg2)
```

Рис. 5: Рис.7

Авторизируемся в гитхабе на устройстве.

```
[andre@andre Операционные системы]$ gh auth login
? You're already logged into github.com. Do you want to re-authenticate? Yes
? What is your preferred protocol for Git operations? HTTPS
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 815C-3F7D
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as AndrewGaina
```

Рис. 6: Рис.8

Создаем репозиторий на гитхабе с помощью template. Потом клонируем его себе на компьютер.

```
[andre@andre Операционные системы]$ git clone --recursive https://github.com/yamadharma/course-directory-student-template.git
Клонирование в «course-directory-student-template»…
```

Рис. 7: Рис.9

Вносим поправки в репозиторий на компьютере.

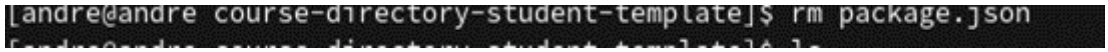

Рис. 8: Рис.10

Добавляем файлы с поправками в коммит и отправляем на сервер.

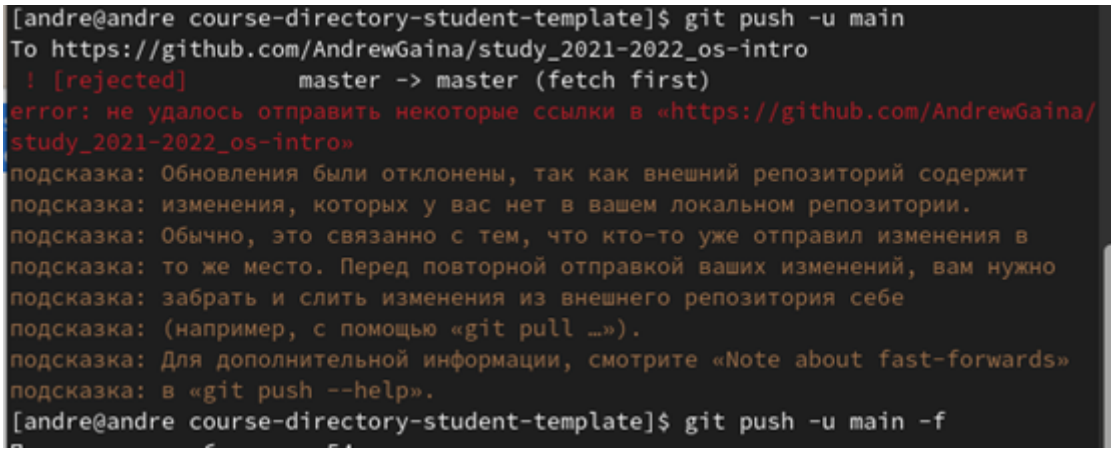

Рис. 9: Рис.11

Убеждаемся что всё работает. Вуаля

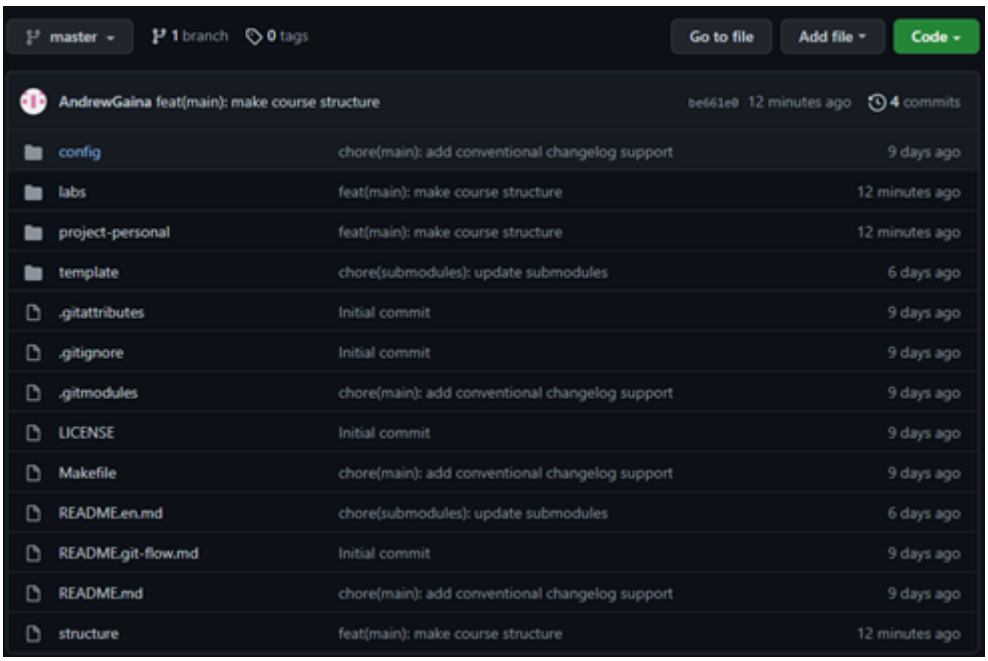
master
1 branch
0 tags
Go to file
Add file
Code
AndrewGaina feat(main): make course structure
be661e0 12 minutes ago
4 commits
config chore(main): add conventional changelog support 9 days ago
labs feat(main): make course structure 12 minutes ago
project-personal feat(main): make course structure 12 minutes ago
template chore(submodules): update submodules 6 days ago
.gitattributes Initial commit 9 days ago
.gitignore Initial commit 9 days ago
.gitmodules chore(main): add conventional changelog support 9 days ago
LICENSE Initial commit 9 days ago
Makefile chore(main): add conventional changelog support 9 days ago
README.en.md chore(submodules): update submodules 6 days ago
README.git-flow.md Initial commit 9 days ago
README.md chore(main): add conventional changelog support 9 days ago
structure feat(main): make course structure 12 minutes ago

Рис. 10: Рис.12

Выводы

Благодаря данной работе мы научились пользоваться гитом.

Контрольные вопросы

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

Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение

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

Хранилище версий - то, где хранятся все документы вместе с историей их изменения и другой служебной информацией. Коммит - зафиксированный набор изменений, который показывает, какие файлы изменились и что именно в них изменилось. История - список всех изменений. Рабочая копия - снимок одной версии проекта

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

Централизованные: одно основное хранилище всего проекта; каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно. Примеры: Subversion, CVS, TFS, VAULT, AccuRev. Децентрализованные: у каждого пользователя свой вариант (возможно не один) репозитория, присутствует возможность добавлять и забирать изменения из любого репозитория. Пример: Git, Mercurial, Bazaar.

Опишите действия с VCS при единоличной работе с хранилищем.

Действия не отличаются от действий при групповой работе: вносим измене-

ния, отправляем новую версию.

Опишите порядок работы с общим хранилищем VCS.

Клонировать репозиторий себе в гитхаб Клонировать репозиторий себе на устройство Внести изменения Добавить новую версию файлов на сервер

Каковы основные задачи, решаемые инструментальным средством git?

Git решает две задачи: хранить информацию о всех изменениях в коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

Назовите и дайте краткую характеристику командам git.

git config - настройки git init - создание репозитория git add - добавление файлов в индекс git commit - коммит изменений git status - список измененных файлов git push - перенос изменений в главную ветку git rm - удаление файлов из индекса

Приведите примеры использования при работе с локальным и удалённым репозиториями.

Локальный репозиторий можно загрузить на гитхаб и работать с ним с помощью VCS, т.е. загружать новые версии, не теряя старые.

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

Ветка в Git это подвижный указатель на один из коммитов. Обычно ветка указывает на последний коммит в цепочке коммитов. Ветки нужны для того, чтобы программисты могли вести совместную работу над проектом и не мешать друг другу при этом.

Как и зачем можно игнорировать некоторые файлы при commit?

Чтобы проигнорировать файлы при коммит, надо просто не добавлять их в коммит. Игнорируют те файлы, которые пользователь не хочет отправлять в репозиторий.