Отчёт о выполнении лабораторной работы №2 Управление версиями

Российский Университет Дружбы Народов

Факульткт Физико-Математических и Естественных Наук

Дисциплина: Операционные системы

Работу выполняла: Адоле Фейт

1032205074

НПМбд-01-20

Москва. Дисплейный класс РУДН. 2021г.

Цель работы

Изучить идеологию и применение средств контроля версий.

Задание

1. Создайте учётную запись на https://github.com.
2. Настройте систему контроля версий git, как это описано выше с использованием сервера репозиториев https://github.com/.
3. Создайте структуру каталога лабораторных работ согласно пункту М.2.
4. Подключение репозитория к github

– Создайте репозиторий на GitHub. Для примера назовём его os-intro.

– Рабочий каталог будем обозначать как laboratory. Вначале нужно перейти в этот каталог:

– Инициализируем системы git:

– Создаём заготовку для файла README.md:

echo “# Лабораторные работы” >> README.md

git add README.md

– Делаем первый коммит и выкладываем на github: git commit -m “first commit”

git remote add origin

↪ git@github.com:/sciproc-intro.git 5. Первичная конфигурация

– Добавим файл лицензии: wget https://creativecommons.org/licenses/by/4.0/legalcode.txt

– Добавим шаблон игнорируемых файлов. Просмотрим список имеющихся шаблонов: curl -L -s https://www.gitignore.io/api/list

Затем скачаем шаблон, например, для C:

curl -L -s https://www.gitignore.io/api/c >> .gitignore

Можно это же сделать через web-интерфейс на сайте https://www.gitignore. io/. – Добавим новые файлы:

git add .

– Выполним коммит: git commit -a

– Отправим на github: git push

6. Конфигурация git-flow

– Инициализируем git-flow git flow init

Префикс для ярлыков установим в v.

– Проверьте, что Вы на ветке develop: git branch

– Создадим релиз с версией 1.0.0

git flow release start 1.0.0

– Запишем версию: echo “1.0.0” >> VERSION

– Добавим в индекс:

git add .

git commit -am ‘chore(main): add version’

– Зальём релизную ветку в основную ветку git flow release finish 1.0.0

– Отправим данные на github

git push –all

git push –tags

– Создадим релиз на github.

Выполнение работы

1. Создала учётную запись на https://github.com.
2. Настройте систему контроля версий git добавила ssh ключ

– Создала репозиторий на GitHub и файл README.md.

3. Первичная конфигурация

– Добавила файл лицензии:

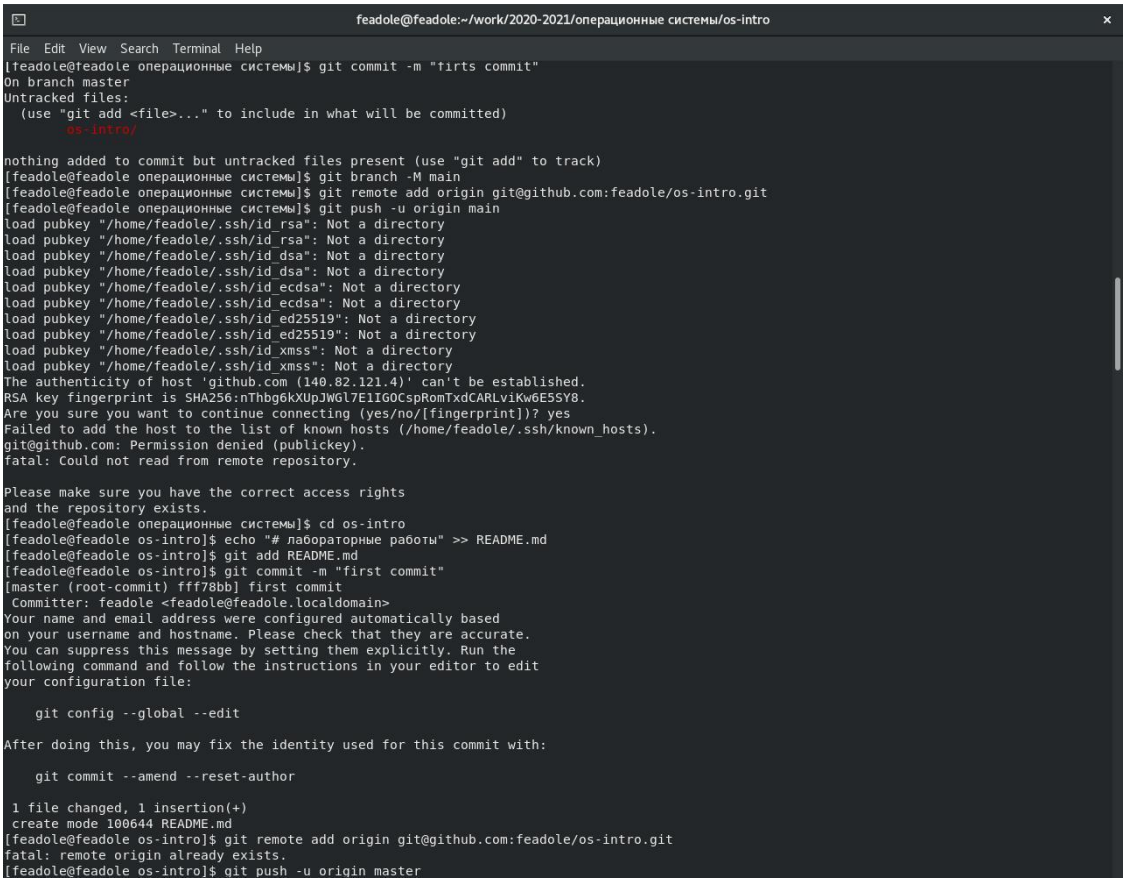
– Добавила шаблон игнорируемых файлов. Просмотрела список имеющихся шаблонов: Затем скачала шаблон, например, для C++: Добавила новые файлы, Выполнила коммит. Отправила на github.

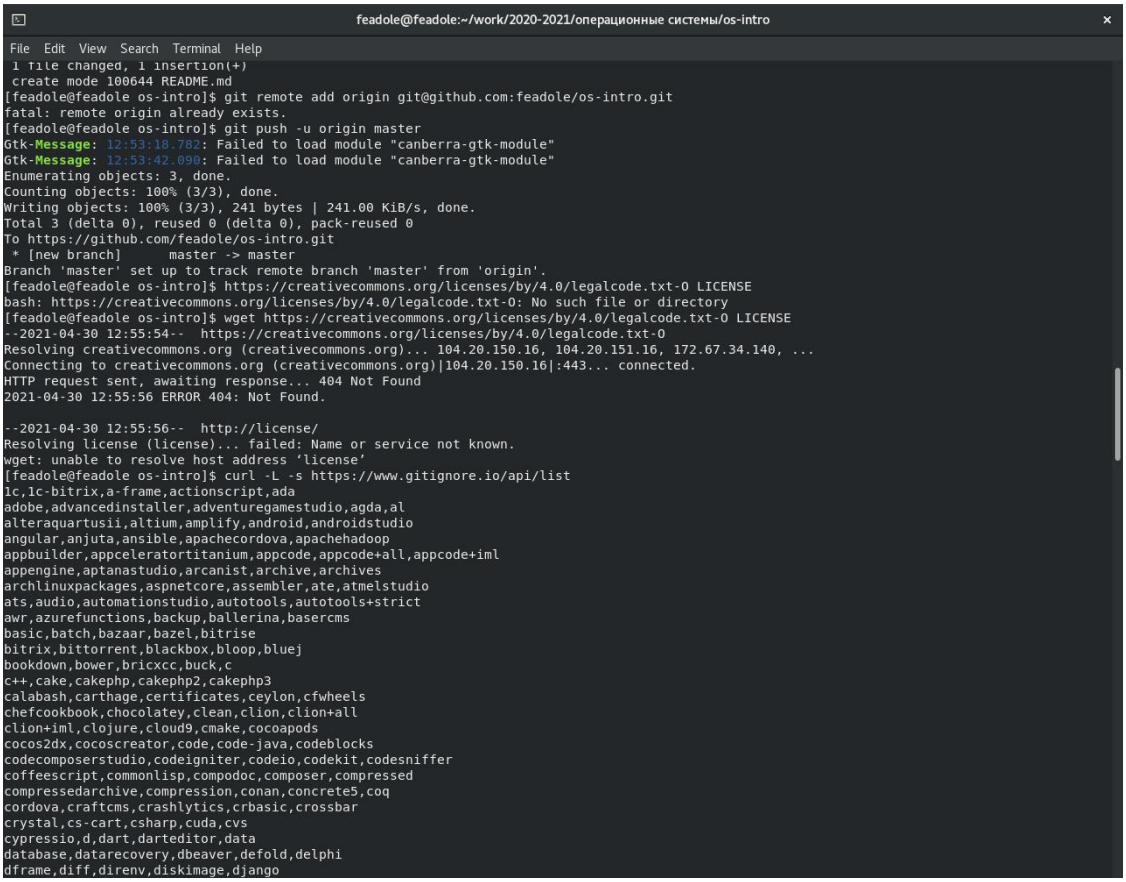
4. Конфигурация git-flow Инициализировала git-flow Проверила, что я на ветке develop: Создала релиз с версией 1.0.0 Записала версию:

Добавила в индекс:

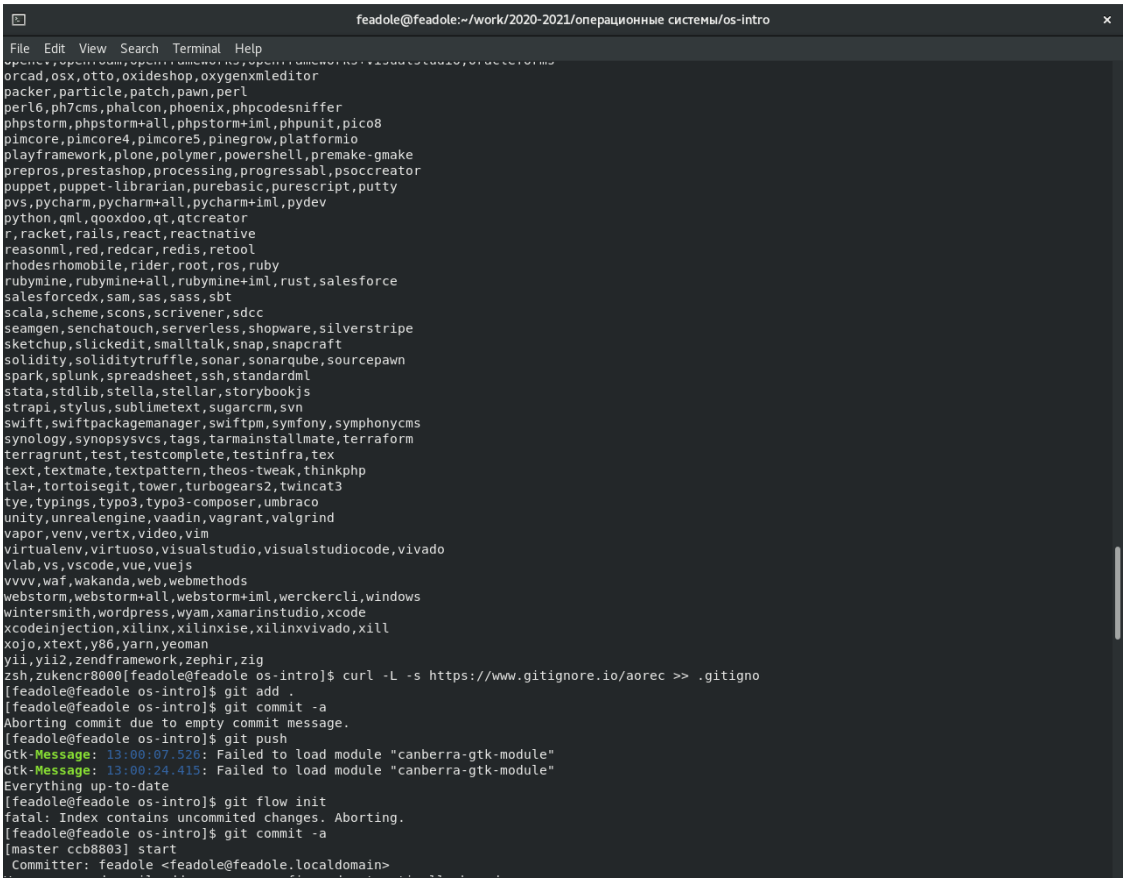
Залила релизную ветку в основную ветку Отправила данные на github

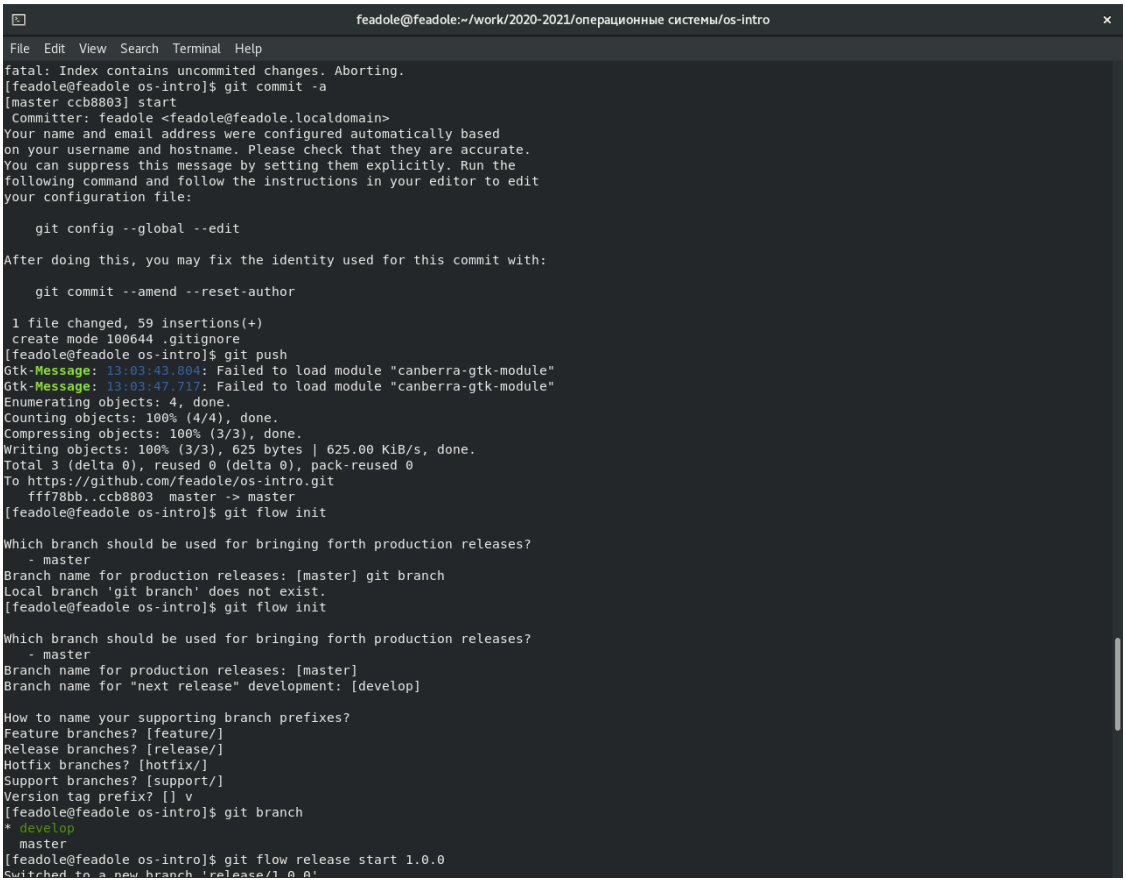
Создадим релиз на github.

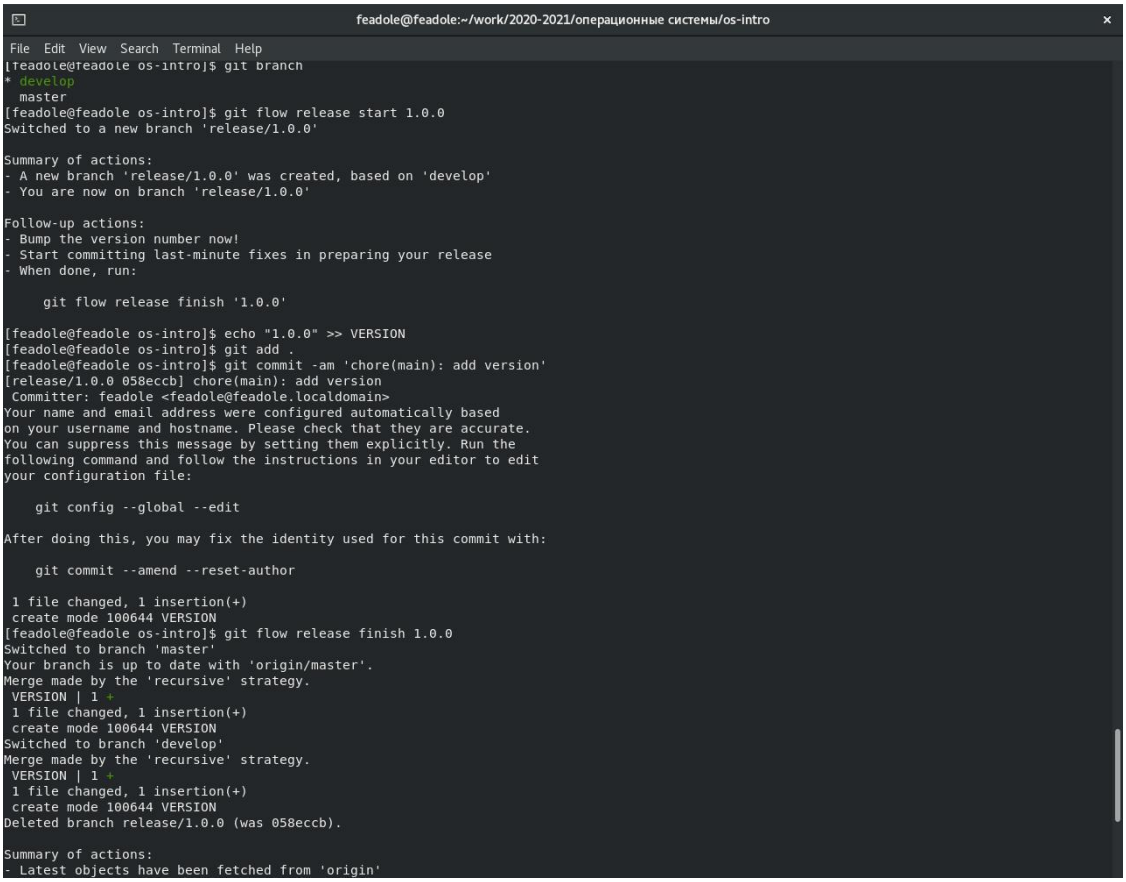

feadole@feadole:~/work/2020-2021/операционные системы/os-intro
File Edit View Search Terminal Help
[feadole@feadole операционные системы]$ git commit -m "firts commit"
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        os-intro/
nothing added to commit but untracked files present (use "git add" to track)
[feadole@feadole операционные системы]$ git branch -M main
[feadole@feadole операционные системы]$ git remote add origin git@github.com:feadole/os-intro.git
[feadole@feadole операционные системы]$ git push -u origin main
load pubkey "/home/feadole/.ssh/id_rsa": Not a directory
load pubkey "/home/feadole/.ssh/id_rsa": Not a directory
load pubkey "/home/feadole/.ssh/id_dsa": Not a directory
load pubkey "/home/feadole/.ssh/id_dsa": Not a directory
load pubkey "/home/feadole/.ssh/id_ecdsa": Not a directory
load pubkey "/home/feadole/.ssh/id_ecdsa": Not a directory
load pubkey "/home/feadole/.ssh/id_ed25519": Not a directory
load pubkey "/home/feadole/.ssh/id_ed25519": Not a directory
load pubkey "/home/feadole/.ssh/id_xmss": Not a directory
load pubkey "/home/feadole/.ssh/id_xmss": Not a directory
The authenticity of host 'github.com (140.82.121.4)' can't be established.
RSA key fingerprint is SHA256:nThbg6kXUpJWGl7E1IGOCspRomTxdCARLviKw6E5SY8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Failed to add the host to the list of known hosts (/home/feadole/.ssh/known_hosts).
git@github.com: Permission denied (publickey).
fatal: Could not read from remote repository.
Please make sure you have the correct access rights
and the repository exists.
[feadole@feadole операционные системы]$ cd os-intro
[feadole@feadole os-intro]$ echo "# лабораторные работы" >> README.md
[feadole@feadole os-intro]$ git add README.md
[feadole@feadole os-intro]$ git commit -m "first commit"
[master (root-commit) fff78bb] first commit
 Committer: feadole <feadole@feadole.localdomain>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:
    git config --global --edit
After doing this, you may fix the identity used for this commit with:
    git commit --amend --reset-author
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
[feadole@feadole os-intro]$ git remote add origin git@github.com:feadole/os-intro.git
fatal: remote origin already exists.
[feadole@feadole os-intro]$ git push -u origin master


feadole@feadole:~/work/2020-2021/операционные системы/os-intro
File Edit View Search Terminal Help
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
[feadole@feadole os-intro]$ git remote add origin git@github.com:feadole/os-intro.git
fatal: remote origin already exists.
[feadole@feadole os-intro]$ git push -u origin master
Gtk-Message: 12:53:18.782: Failed to load module "canberra-gtk-module"
Gtk-Message: 12:53:42.090: Failed to load module "canberra-gtk-module"
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 241 bytes | 241.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/feadole/os-intro.git
 * [new branch]   master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
[feadole@feadole os-intro]$ https://creativecommons.org/licenses/by/4.0/legalcode.txt-O LICENSE
bash: https://creativecommons.org/licenses/by/4.0/legalcode.txt-O: No such file or directory
[feadole@feadole os-intro]$ wget https://creativecommons.org/licenses/by/4.0/legalcode.txt-O LICENSE
--2021-04-30 12:55:54-- https://creativecommons.org/licenses/by/4.0/legalcode.txt-O
Resolving creativecommons.org (creativecommons.org)... 104.20.150.16, 104.20.151.16, 172.67.34.140, ...
Connecting to creativecommons.org (creativecommons.org)|104.20.150.16|:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2021-04-30 12:55:56 ERROR 404: Not Found.

--2021-04-30 12:55:56-- http://license/
Resolving license (license)... failed: Name or service not known.
wget: unable to resolve host address ‘license’
[feadole@feadole os-intro]$ curl -L -s https://www.gitignore.io/api/list
1c,1c-bitrix,a-frame,actionscript,ada
adobe,advancedinstaller,adventuregamestudio,agda,al
alteraquartusii,altium,amplify,android,androidstudio
angular,anjuta,ansible,apachecordova,apachehadoop
appbuilder,appceleratortitanium,appcode,appcode+all,appcode+iml
appengine,aptanastudio,arcanist,archive,archives
archlinuxpackages,aspnetcore,assembler,ate,atmelstudio
ats,audio,automationstudio,autotools,autotools+strict
awr,azurefunctions,backup,ballerina,basercms
basic,batch,bazaar,bazel,bitrise
bitrix,bittorrent,blackbox,bloop,bluej
bookdown,bower,bricxcc,buck,c
c++,cake,cakephp,cakephp2,cakephp3
calabash,carthage,certificates,ceylon,cfwheels
chefcookbook,chocolatey,clean,clion,clion+all
clion+iml,clojure,cloud9,cmake,cocoapods
cocos2dx,cocoscreator,code,code-java,codeblocks
codecomposerstudio,codeigniter,codeio,codekit,codesniffer
coffeescript,commonlisp,compodoc,composer,compressed
compressedarchive,compression,conan,concrete5,coq
cordova,craftcms,crashlytics,crbasic,crossbar
crystal,cs-cart,csharp,cuda,cvs
cypressio,d,dart,darteditor,data
database,datarecovery,dbeaver,defold,delphi
dframe,diff,direnv,diskimage,django


feadole@feadole:~/work/2020-2021/операционные системы/os-intro
File Edit View Search Terminal Help
orcad,osx,otto,oxideshop,oxygenxmleditor
packer,particle,patch,pawn,perl
perl6,ph7cms,phalcon,phoenix,phpcodesniffer
phpstorm,phpstorm+all,phpstorm+iml,phpunit,pico8
pimcore,pimcore4,pimcore5,pinegrow,platformio
playframework,plone,polymer,powershell,premake-gmake
prepros,prestashop,processing,progressabl,psoccreator
puppet,puppet-librarian,purebasic,purescript,putty
pvs,pycharm,pycharm+all,pycharm+iml,pydev
python,qml,qooxdoo,qt,qtcreator
r,racket,rails,react,reactnative
reasonml,red,redcar,redis,retool
rhodesrhomobile,rider,root,ros,ruby
rubymine,rubymine+all,rubymine+iml,rust,salesforce
salesforcedx,sam,sas,sass,sbt
scala,scheme,scons,scrivener,sdcc
seamgen,senchatouch,serverless,shopware,silverstripe
sketchup,slickedit,smalltalk,snap,snapcraft
solidity,soliditytruffle,sonar,sonarqube,sourcepawn
spark,splunk,spreadsheet,ssh,standardml
stata,stdlib,stella,stellar,storybookjs
strapi,stylus,sublimetext,sugarcrm,svn
swift,swiftpackagemanager,swiftpm,symfony,symphonycms
synology,synopsysvcs,tags,tarmainstallmate,terraform
terragrunt,test,testcomplete,testinfra,tex
text,textmate,textpattern,theos-tweak,thinkphp
tla+,tortoisegit,tower,turbogears2,twincat3
tye,typings,typo3,typo3-composer,umbraco
unity,unrealengine,vaadin,vagrant,valgrind
vapor,venv,vertx,video,vim
virtualenv,virtuoso,visualstudio,visualstudiocode,vivado
vlab,vs,vscode,vue,vuejs
vvvv,waf,wakanda,web,webmethods
webstorm,webstorm+all,webstorm+iml,werckercli,windows
wintersmith,wordpress,wyam,xamarinstudio,xcode
xcodeinjection,xilinx,xilinxise,xilinxvivado,xill
xojo,xtext,y86,yarn,yeoman
yii,yii2,zendframework,zephir,zig
zsh,zukencr8000[feadole@feadole os-intro]$ curl -L -s https://www.gitignore.io/aorec >> .gitigno
[feadole@feadole os-intro]$ git add .
[feadole@feadole os-intro]$ git commit -a
Aborting commit due to empty commit message.
[feadole@feadole os-intro]$ git push
Gtk-Message: 13:00:07.526: Failed to load module "canberra-gtk-module"
Gtk-Message: 13:00:24.415: Failed to load module "canberra-gtk-module"
Everything up-to-date
[feadole@feadole os-intro]$ git flow init
fatal: Index contains uncommited changes. Aborting.
[feadole@feadole os-intro]$ git commit -a
[master ccb8803] start
Committer: feadole <feadole@feadole.localdomain>


feadole@feadole:~/work/2020-2021/операционные системы/os-intro
File Edit View Search Terminal Help
fatal: Index contains uncommited changes. Aborting.
[feadole@feadole os-intro]$ git commit -a
[master ccb8803] start
 Committer: feadole <feadole@feadole.localdomain>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:
git config --global --edit
After doing this, you may fix the identity used for this commit with:
git commit --amend --reset-author
1 file changed, 59 insertions(+)
create mode 100644 .gitignore
[feadole@feadole os-intro]$ git push
Gtk-Message: 13:03:43.804: Failed to load module "canberra-gtk-module"
Gtk-Message: 13:03:47.717: Failed to load module "canberra-gtk-module"
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 625 bytes | 625.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/feadole/os-intro.git
fff78bb..ccb8803 master -> master
[feadole@feadole os-intro]$ git flow init
Which branch should be used for bringing forth production releases?
- master
Branch name for production releases: [master] git branch
Local branch 'git branch' does not exist.
[feadole@feadole os-intro]$ git flow init
Which branch should be used for bringing forth production releases?
- master
Branch name for production releases: [master]
Branch name for "next release" development: [develop]
How to name your supporting branch prefixes?
Feature branches? [feature/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? [] v
[feadole@feadole os-intro]$ git branch
* develop
master
[feadole@feadole os-intro]$ git flow release start 1.0.0
Switched to a new branch 'release/1.0.0'



контрольные вопросы

1. 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. В Git один коммит (англ. commit) представляет из себя ссылку на объект tree, соответствующий корневой директории, и ссылку на родительский коммит (кроме самого первого коммита в репозитории). Также в коммите есть информация об авторе и UNIX timestamp от времени создания. В Git единицей хранения данных является объект (англ. object), который однозначно определяется 40-символьным хешем sha1. В объектах Git хранит почти всё: коммиты, содержимое файлов, их иерархию. Сначала объекты представляют из себя обычные файлы в папке .git/objects, а после git gc упаковываются в .pack-файлы, о которых будет рассказано чуть ниже. Для экономии дискового пространства содержимое всех объектов дополнительно сжимается с помощью zlib. В Git нет отдельного хранилища истории. Всю историю можно развернуть, но лишь пройдя по ссылкам на родителя из нужного вам коммита.

Если необходимо просмотреть историю только по одному файлу (или по поддиректории), Git всё равно должен проделать то же самое, но он будет возвращать отфильтрованные результаты. Стоит иметь это ввиду, когда вы делаете интеграцию с Git, и не заставлять Git делать полный просмотр истории на каждый файл. Рабочая копия является снимком одной версии проекта. Эти файлы извлекаются из сжатой базы данных в каталоге Git и помещаются на диск, для того чтобы их можно было использовать или редактировать.

3. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

4. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент.

5. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент.

6. Задачи решаемые git: Как не потерять файлы с исходным кодом? Как защититься от случайных исправлений и удалений? Как отменить изменения, если они оказались некорректными? Как одновременно поддерживать рабочую версию и разработку новой?

7. – создание основного дерева репозитория: git init

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

git pull

– отправка всех произведённых изменений локального дерева в центральный репозиторий: git push

– просмотр списка изменённых файлов в текущей директории:

git status

– просмотр текущих изменения: git diff

– сохранение текущих изменений:

– добавить все изменённые и/или созданные файлы и/или каталоги: git add .

– добавить конкретные изменённые и/или созданные файлы и/или каталоги: git add имена_файлов

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

git rm имена_файлов

– сохранение добавленных изменений:

– сохранить все добавленные изменения и все изменённые файлы:

git commit -am ‘Описание коммита’

– сохранить добавленные изменения с внесением комментария через встроенный редактор: git commit

8. git push –all
9. Ветви функциональностей (feature branches), также называемые иногда тематическими ветвями (topic branches), используются для разработки новых функций, которые должны появиться в текущем или будущем релизах.
10. Игнорируемые файлы – это, как правило, специфичные для платформы файлы или автоматически созданные файлы из систем сборки. Некоторые общие примеры включают в себя: Файлы времени выполнения, такие как журнал, блокировка, кэш или временные файлы. Файлы с конфиденциальной информацией, такой как пароли или ключи API. Скомпилированный код, такой как .class или .o.

Каталоги зависимостей, такие как /vendor или /node_modules. Создавать папки, такие как /public, /out или /dist.

Системные файлы, такие как .DS_Store или Thumbs.db Конфигурационные файлы IDE или текстового редактора.

Вывод

Изучила идеологию и применение средств контроля версий. Создала аккаунт и репозиторий на github. Выполнила важнейшие команды.