

Отчёт по лабораторной работе №2

Управление версиями

Цзян Вэньцзе

Содержание

| | | |
|----------|---------------------------------------|-----------|
| 1 | Цель работы | 5 |
| 2 | Выполнение лабораторной работы | 6 |
| 3 | Вывод | 17 |
| 4 | Контрольные вопросы | 18 |

Список иллюстраций

| | | |
|------|---|----|
| 2.1 | Загрузка пакетов | 7 |
| 2.2 | Параметры репозитория | 8 |
| 2.3 | rsa-4096 | 9 |
| 2.4 | ed25519 | 10 |
| 2.5 | GPG ключ | 11 |
| 2.6 | GPG ключ | 12 |
| 2.7 | Параметры репозитория | 13 |
| 2.8 | Связь репозитория с аккаунтом | 14 |
| 2.9 | Загрузка шаблона | 15 |
| 2.10 | Первый коммит | 16 |

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.

```
jiangwenjie@jiangwenjie:~$ git
использование: 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] [--no-lazy-fetch]
                [--no-optional-locks] [--no-advice] [--bare] [--git-dir=<path>]
                [--work-tree=<path>] [--namespace=<name>] [--config-env=<name>=<envvar>]
                <command> [<args>]
```

Стандартные команды Git используются в различных ситуациях:

создание рабочей области (смотрите также: `git help tutorial`)

| | |
|--------------------|--|
| <code>clone</code> | Клонирование репозитория в новый каталог |
| <code>init</code> | Создание пустого репозитория Git или переинициализация существующего |

работа с текущими изменениями (смотрите также: `git help everyday`)

| | |
|----------------------|---|
| <code>add</code> | Добавление содержимого файла в индекс |
| <code>mv</code> | Перемещение или переименование файла, каталога или символической ссылки |
| <code>restore</code> | Восстановление файлов в рабочем каталоге |
| <code>rm</code> | Удаление файлов из рабочего каталога и индекса |

просмотр истории и текущего состояния (смотрите также: `git help revisions`)

| | |
|---------------------|--|
| <code>bisect</code> | Выполнение двоичного поиска коммита, который вносит ошибку |
| <code>diff</code> | Вывод разницы между коммитами, коммитом и рабочим каталогом и т.д. |
| <code>grep</code> | Вывод строк, соответствующих шаблону |
| <code>log</code> | Вывод истории коммитов |
| <code>show</code> | Вывод различных типов объектов |
| <code>status</code> | Вывод состояния рабочего каталога |

выращивание, маркировка и правка вашей общей истории

| | |
|---------------------|--|
| <code>branch</code> | Вывод списка, создание или удаление веток |
| <code>commit</code> | Запись изменений в репозиторий |
| <code>merge</code> | Объединение одной или нескольких историй разработки вместе |
| <code>rebase</code> | Повторное применение коммитов над верхушкой другой ветки |
| <code>reset</code> | Сброс текущего состояния HEAD на указанное состояние |
| <code>switch</code> | Переключение веток |
| <code>tag</code> | Создание, вывод списка, удаление или проверка метки, подписанной с помощью GPG |

Рис. 2.1: Загрузка пакетов

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

```
j'iangwenjie@j'iangwenjie:~$  
j'iangwenjie@j'iangwenjie:~$ git config --global user.name "j'iangwenjie8724"  
j'iangwenjie@j'iangwenjie:~$ git config --global user.email "1632234424@qq.ru"  
j'iangwenjie@j'iangwenjie:~$ git config --global core.quotepath false  
j'iangwenjie@j'iangwenjie:~$ git config --global init.defaultBranch master  
j'iangwenjie@j'iangwenjie:~$ git config --global core.autocrlf input  
j'iangwenjie@j'iangwenjie:~$ git config --global core.safecrlf warn  
j'iangwenjie@j'iangwenjie:~$
```

Рис. 2.2: Параметры репозитория

Создаем SSH ключи


```

jiangwenjie@jiangwenjie:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/jiangwenjie/.ssh/id_rsa):
Created directory '/home/jiangwenjie/.ssh'.
Enter passphrase for "/home/jiangwenjie/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jiangwenjie/.ssh/id_rsa
Your public key has been saved in /home/jiangwenjie/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Kok47ovqLU2Fpgur2CynrX0Bi5TyB4JoHzLtNQ9cHOM jiangwenjie@jiangwenjie
The key's randomart image is:
+---[RSA 4096]-----+
|      .o.      |
|      .o.      |
|o o o .E       |
|=*o= *        |
|++B++ + S     |
|ooo=o. o      |
|+.=.o..       |
|*Oo...        |
| %XBo         |
+---[SHA256]-----+
jiangwenjie@jiangwenjie:~$

```

Рис. 2.3: rsa-4096

```

jiangwenjie@jiangwenjie:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/jiangwenjie/.ssh/id_ed25519):
Enter passphrase for "/home/jiangwenjie/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jiangwenjie/.ssh/id_ed25519
Your public key has been saved in /home/jiangwenjie/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:224WVocmTB0GxtA+8a9ngTzNVUcnIFZTfWbYLHDYeTA jiangwenjie@jiangwenjie
The key's randomart image is:
+--[ED25519 256]--+
|      .+ .o=*EB |
|      *oo..oB.X|
|      o = . o B.|
|      o + * o |
|      So X + |
|      o* = |
|      ...+ . |
|      .+ o |
|      o.o |
+----[SHA256]-----+
jiangwenjie@jiangwenjie:~$ █

```

Рис. 2.4: ed25519

Создаем GPG ключ

```

GnuPG должен составить идентификатор пользователя для идентификации ключа.

Ваше полное имя: jiangwenjie0724
Адрес электронной почты: 1032234424@pfur.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
    "jiangwenjie0724 <1032234424@pfur.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? O
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: /home/jiangwenjie/.gnupg/trustdb.gpg: создана таблица доверия
gpg: создан каталог '/home/jiangwenjie/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/home/jiangwenjie/.gnupg/openpgp-revocs.d/5DA8134223CEA839D5E89DFA58749FF9251A550E.rev'.
открытый и секретный ключи созданы и подписаны.

pub  rsa4096 2025-09-04 [SC]
      5DA8134223CEA839D5E89DFA58749FF9251A550E
uid                               jiangwenjie0724 <1032234424@pfur.ru>
sub   rsa4096 2025-09-04 [E]

jiangwenjie@jiangwenjie:~$

```

Рис. 2.5: GPG ключ


Добавляем GPG ключ в аккаунт

SSH keys

[New SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication keys

**jiang**
SHA256: 8p0Sr0w7bG+F17it41Ine14bz6Cjyo1MQmwYxKm8kE
Added on Mar 11, 2025
Never used — Read/write


Delete

Check out our guide to [connecting to GitHub using SSH keys](#) or troubleshoot [common SSH problems](#).

GPG keys

[New GPG key](#)

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.

**2**
Email address: 1032234424@pfur.ru
Key ID: 58749FF9251A550E
Subkeys: 658EEBA5EE0B4FD3
Added on Sep 4, 2025

Delete

Learn how to [generate a GPG key and add it to your account](#).

Рис. 2.6: GPG ключ

Настройка автоматических подписей коммитов git

```

jiangwenjie@jiangwenjie:~$
jiangwenjie@jiangwenjie:~$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1u
[keyboard]
-----
sec  rsa4096/58749FF9251A550E 2025-09-04 [SC]
      5DA8134223CEA839D5E89DFA58749FF9251A550E
uid          [ абсолютно ] jiangwenjie0724 <1032234424@pfur.ru>
ssb  rsa4096/658EEBA5EE0B4FD3 2025-09-04 [E]

jiangwenjie@jiangwenjie:~$ gpg --armor --export 58749FF9251A550E | xclip -sel clip
jiangwenjie@jiangwenjie:~$
jiangwenjie@jiangwenjie:~$ git config --global user.signingkey 58749FF9251A550E
jiangwenjie@jiangwenjie:~$ git config --global commit.gpgsign true
jiangwenjie@jiangwenjie:~$ git config --global gpg.program $(which gpg2)
jiangwenjie@jiangwenjie:~$

```

Рис. 2.7: Параметры репозитория

Настройка gh

```
jiangwenjie@jiangwenjie:~$  
jiangwenjie@jiangwenjie:~$ gh auth login  
? Where do you use GitHub? GitHub.com  
? What is your preferred protocol for Git operations on this host? SSH  
? Upload your SSH public key to your GitHub account? /home/jiangwenjie/.ssh/id_rsa.pub  
? Title for your SSH key: GitHub CLI  
? How would you like to authenticate GitHub CLI? Login with a web browser  
  
! First copy your one-time code: 6FFB-A9CE  
Press Enter to open https://github.com/login/device in your browser...  
✓ Authentication complete.  
- gh config set -h github.com git_protocol ssh  
✓ Configured git protocol  
✓ Uploaded the SSH key to your GitHub account: /home/jiangwenjie/.ssh/id_rsa.pub  
✓ Logged in as jiangwenjie0724  
jiangwenjie@jiangwenjie:~$  
jiangwenjie@jiangwenjie:~$
```

Рис. 2.8: Связь репозитория с аккаунтом

Загрузка шаблона репозитория и синхронизация

```

jiangwenjie@jiangwenjie:~$ mkdir -p ~/work/study/2024-2025/"Операционные системы"
jiangwenjie@jiangwenjie:~$ cd ~/work/study/2024-2025/"Операционные системы"
jiangwenjie@jiangwenjie:~/work/study/2024-2025/Операционные системы$ gh repo create os-intro --template=yamadharma/course-directory-student-template --public
✓ Created repository jiangwenjie0724/os-intro on GitHub
https://github.com/jiangwenjie0724/os-intro
jiangwenjie@jiangwenjie:~/work/study/2024-2025/Операционные системы$ git clone --recursive git@github.com:jiangwenjie0724/os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TzJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 38, done.
remote: Counting objects: 100% (38/38), done.
remote: Compressing objects: 100% (36/36), done.
remote: Total 38 (delta 1), reused 26 (delta 1), pack-reused 0 (from 0)
Получение объектов: 100% (38/38), 20.68 Киб | 6.89 Миб/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/jiangwenjie/work/study/2024-2025/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 147, done.
remote: Counting objects: 100% (147/147), done.
remote: Compressing objects: 100% (100/100), done.
remote: Total 147 (delta 55), reused 131 (delta 39), pack-reused 0 (from 0)
Получение объектов: 100% (147/147), 2.64 Миб | 8.78 Миб/с, готово.
Определение изменений: 100% (55/55), готово.
Клонирование в «/home/jiangwenjie/work/study/2024-2025/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 207, done.
remote: Counting objects: 100% (207/207), done.
remote: Compressing objects: 100% (141/141), done.
remote: Total 207 (delta 92), reused 170 (delta 55), pack-reused 0 (from 0)
Получение объектов: 100% (207/207), 760.70 Киб | 3.62 Миб/с, готово.
Определение изменений: 100% (92/92), готово.
Submodule path 'template/presentation': checked out '645759e4b104e93753637dedf8109adf24d071b7'
Submodule path 'template/report': checked out 'cbca6e80e9c8e9b190a4bbbb8595f82335ae634a'
jiangwenjie@jiangwenjie:~/work/study/2024-2025/Операционные системы$

```

Рис. 2.9: Загрузка шаблона

Подготовка репозитория и коммит изменений

```

jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы$
jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы$
jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы$ cd ~/work/study/2024-2025/"Операционные системы"/os-intro
0
jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы/os-intro$ make COURSE=os-intro prepare
jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы/os-intro$ rm package.json
jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы/os-intro$ ls
COURSE  LICENSE  prepare  project-personal  README.git-flow.md  template
labs    Makefile  presentation  README.en.md      README.md
jjiangwenjie@jjiangwenjie:~/work/study/2024-2025/Операционные системы/os-intro$

```

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: