

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**
**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

**«Исследование возможностей Git для работы с
локальными репозиториями»**

Отчет по лабораторной работе № 1.2

по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-21-1

Пуценко И. А. .« » сентября 2022г.

Подпись студента _____

Работа защищена « » _____ 20__г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2022

Методика и порядок выполнения работы

1. Отработка примеров лабораторной работы.

Запуск команд `git log`, `git log -p -2`, `git log --stat`, `git log --pretty=oneline`, `git log --pretty=short`, `git log --format :"%h - %s, %an , %ar"`, `git log --pretty=format:"%h %s" --graph`, `git log --since=2.weeks`, `git log -S function_name` в каталоге клонированного проекта, показан на рисунках 1,2,3,4,5,6,7,8,9,.

```
C:\Users\FonK\Desktop\ОПИ>cd simplegit-progit
C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log
commit ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD)
Author: Scott Chacon <schacon@gmail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the verison number

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 10:31:28 2008 -0700

    first commit
```

Рисунок 1 – Результат команды `git log`

```

C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log -p -2
commit ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD)
Author: Scott Chacon <schacon@gmail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the verison number

diff --git a/Rakefile b/Rakefile
index a874b73..8f94139 100644
--- a/Rakefile
+++ b/Rakefile
@@ -5,7 +5,7 @@ require 'rake/gempackagetask'
 spec = Gem::Specification.new do |s|
   s.platform = Gem::Platform::RUBY
   s.name      = "simplegit"
-  s.version   = "0.1.0"
+  s.version   = "0.1.1"
   s.author    = "Scott Chacon"
   s.email     = "schacon@gmail.com"
   s.summary   = "A simple gem for using Git in Ruby code."

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index a0a60ae..47c6340 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -18,8 +18,3 @@ class SimpleGit
   end

   end

-  -if $0 == __FILE__
-    git = SimpleGit.new
-    puts git.show
-  end
\ No newline at end of file

```

Рисунок 2 – Результат команды git log -p -2

```

C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log --stat
commit ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD)
Author: Scott Chacon <schacon@gmail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the verison number

Rakefile | 2 +-
1 file changed, 1 insertion(+), 1 deletion(-)

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

lib/simplegit.rb | 5 -----
1 file changed, 5 deletions(-)

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gmail.com>
Date:   Sat Mar 15 10:31:28 2008 -0700

    first commit

README           | 6 ++++++
Rakefile         | 23 +++++++++++++++++++++++++++++++++++++
lib/simplegit.rb | 25 +++++++++++++++++++++++++++++++++++++
3 files changed, 54 insertions(+)

```

Рисунок 3 – Результат команды git log --stat

```

C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log --pretty=oneline
ca82a6dff817ec66f44342007202690a93763949 (HEAD -> master, origin/master, origin/HEAD) changed the verison number
085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7 removed unnecessary test code
a11bef06a3f659402fe7563abf99ad00de2209e6 first commit

C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log --pretty=format:"%h %s" --graph
* ca82a6d changed the verison number
* 085bb3b removed unnecessary test code
* a11bef0 first commit

```

Рисунок 4 – Результат команды git log --pretty=oneline

```

C:\Users\FonK\Desktop\ОПИ\labRab_2>git log --pretty=short
commit ddae3afd4257e9c9575e92d0da9cd0cdf942be32 (HEAD -> main, tag: v1.2)
Author: Chmonya <towerland2801@gmail.com>

    commit v1.2

commit 635d0bb8d5c49acf0844c685e2db02f4c981b3aa
Author: Chmonya <towerland2801@gmail.com>

    change main file

commit b2462fef706dd5a52591539d0602fbd4001156a2
Author: Chmonya <towerland2801@gmail.com>

    change main file

commit 366725512b351bdc7d505bf27cbff268eb6283cc (tag: v1.1)
Author: Chmonya <towerland2801@gmail.com>

    chenge main file 1.1

commit b1229db2b4df6091ad4e9ee86b2441650a7be023
Author: Chmonya <towerland2801@gmail.com>

    chenge main file

commit fa7253b434b6659b5acbf1759d8c5750d46ea77c (tag: v1.0)
Author: Chmonya <towerland2801@gmail.com>

    created main file

commit 436d69b6999295b451340e8b4b478542a1541aa2
Author: kvakaet <118031391+kvakaet@users.noreply.github.com>

    Initial commit

```

Рисунок 5 – Результат команды git log --pretty=short

```
C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log --pretty=format:"%h - %an, %ar : %s"
ca82a6d - Scott Chacon, 15 years ago : changed the verison number
085bb3b - Scott Chacon, 15 years ago : removed unnecessary test code
a11bef0 - Scott Chacon, 15 years ago : first commit
```

Рисунок 6 – Результат команды `git log --pretty=format:"%h - %s, %an , %ar"`

```
C:\Users\FonK\Desktop\ОПИ\simplegit-progit>git log --pretty=format:"%h %s" --graph
* ca82a6d changed the verison number
* 085bb3b removed unnecessary test code
* a11bef0 first commit
```

Рисунок 7 – Результат команды `git log --pretty=format:"%h %s" --graph`

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git log --since=2.weeks
commit ddae3afbd4257e9c9575e92d0da9cd0cdf942be32 (HEAD -> main, tag: v1.2)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:30:55 2022 +0300

    commit v1.2

commit 635d0bb8d5c49acf0844c685e2db02f4c981b3aa
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:28:15 2022 +0300

    change main file

commit b2462fef706dd5a52591539d0602fbd4001156a2
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:18:04 2022 +0300

    change main file

commit 366725512b351bdc7d505bf27cbff268eb6283cc (tag: v1.1)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:11:38 2022 +0300

    chenge main file 1.1

commit b1229db2b4df6091ad4e9ee86b2441650a7be023
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:08:55 2022 +0300

    chenge main file

commit fa7253b434b6659b5acbf1759d8c5750d46ea77c (tag: v1.0)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 05:45:44 2022 +0300

    created main file

commit 436d69b6999295b451340e8b4b478542a1541aa2
Author: kvakaet <118031391+kvakaet@users.noreply.github.com>
Date:   Mon Dec 5 05:10:18 2022 +0300

    Initial commit
```

Рисунок 8 – Результат команды git log --since=2.weeks

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git log -S print
commit ddae3afd4257e9c9575e92d0da9cd0cdf942be32 (HEAD -> main, tag: v1.2)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:30:55 2022 +0300

    commit v1.2

commit 635d0bb8d5c49acf0844c685e2db02f4c981b3aa
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:28:15 2022 +0300

    change main file

commit b2462fef706dd5a52591539d0602fbd4001156a2
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:18:04 2022 +0300

    change main file

commit 366725512b351bdc7d505bf27cbff268eb6283cc (tag: v1.1)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:11:38 2022 +0300

    chenge main file 1.1

commit b1229db2b4df6091ad4e9ee86b2441650a7be023
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:08:55 2022 +0300

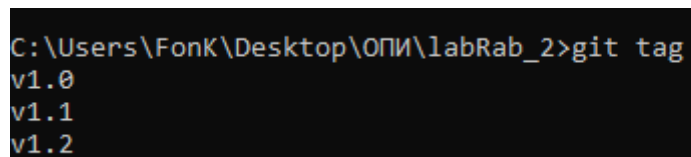
    chenge main file

commit fa7253b434b6659b5acbf1759d8c5750d46ea77c (tag: v1.0)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 05:45:44 2022 +0300

    created main file
```

Рисунок 9 – Результат команды git log -S function_name

Работа с тегами. Команда для просмотра списка тегов `git tag` показана на рисунке 10, для создания и просмотра `git tag -a` на рисунке 11, для обмена тегами `git push origin --tags` на рисунке 12.



```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git tag
v1.0
v1.1
v1.2
```

Рисунок 10 – Результат команды `git tag`

```
C:\Users\FonK\Desktop\ОПИ\simplegit-progit>cd C:\Users\FonK\Desktop\ОПИ\labRab_2
C:\Users\FonK\Desktop\ОПИ\labRab_2>git add .
C:\Users\FonK\Desktop\ОПИ\labRab_2>git commit -m "created main file"
[main fa7253b] created main file
 3 files changed, 6 insertions(+), 1 deletion(-)
 create mode 100644 main.py
C:\Users\FonK\Desktop\ОПИ\labRab_2>git tag -a "v1.0" -m "Version 1.0"
C:\Users\FonK\Desktop\ОПИ\labRab_2>git tag -a v1.0 -m "Version 1.0"
fatal: tag 'v1.0' already exists
C:\Users\FonK\Desktop\ОПИ\labRab_2>git tag
v1.0
```

Рисунок 11 –Создание и просмотр тегов с помощью команд
git tag –a

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git push origin --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 164 bytes | 54.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kvakaet/labRab_2.git
 * [new tag]          v1.2 -> v1.2
```

Рисунок 12 – Результат команды git push origin --tag

Результаты команд git log --graph --pretty=oneline для просмотра истории хранилищ, показаны на рисунке 13.

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git log --graph --pretty=oneline
* ddae3afd4257e9c9575e92d0da9cd0cdf942be32 (HEAD -> main, tag: v1.2, origin/main, origin/HEAD) commit v1.2
* 635d0bb8d5c49acf0844c685e2db02f4c981b3aa change main file
* b2462fef706dd5a52591539d0602fbd4001156a2 change main file
* 366725512b351bdc7d505bf27cbff268eb6283cc (tag: v1.1) change main file 1.1
* b1229db2b4df6091ad4e9ee86b2441650a7be023 change main file
* fa7253b434b6659b5acbf1759d8c5750d46ea77c (tag: v1.0) created main file
* 436d69b6999295b451340e8b4b478542a1541aa2 Initial commit
```

Рисунок 13— результат команды git log --graph --pretty=oneline

Результаты команд git show <ref> , показаны на рисунках 14,15,16 где <ref>

– HEAD: последний коммит;

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git show HEAD
commit ddae3afd4257e9c9575e92d0da9cd0cdf942be32 (HEAD -> main, tag: v1.2, origin/main, origin/HEAD)
Author: Chmonya <towerland2801@gmail.com>
Date: Mon Dec 5 06:30:55 2022 +0300

    commit v1.2

diff --git a/main.py b/main.py
index 55513f4..b653d0c 100644
--- a/main.py
+++ b/main.py
@@ -3,3 +3,4 @@ print("а это уже второй коммит без тега")
 print("а этот коммит уже будет с тегом версии \"1.1\"")
 print("этот коммит опять будет без тега")
 print("а этот коммит на удивление тоже будет без тега")
+print("в этом коммите наконец будет тег")
```

Рисунок 14— результат команды git show HEAD

– HEAD~1 предпоследний коммит (и т. д.);

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git show HEAD~1
commit 635d0bb8d5c49acf0844c685e2db02f4c981b3aa
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:28:15 2022 +0300

    change main file

diff --git a/main.py b/main.py
index 2157ec7..55513f4 100644
--- a/main.py
+++ b/main.py
@@ -2,3 +2,4 @@ print("привет, это первый коммит с тегом \"верси
 print("а это уже второй коммит без тега")
 print("а этот коммит уже будет с тегом версии \"1.1\"")
 print("этот коммит опять будет без тега")
+print("а этот коммит на удивление тоже будет без тега")
```

Рисунок 15— результат команды git show HEAD~1

- 366725512b351bdc7d505bf27cbff268eb6283cc: коммит с указанным хэшем

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git show 366725512b351bdc7d505bf27cbff268eb6283cc
commit 366725512b351bdc7d505bf27cbff268eb6283cc (tag: v1.1)
Author: Chmonya <towerland2801@gmail.com>
Date:   Mon Dec 5 06:11:38 2022 +0300

    chenge main file 1.1

diff --git a/main.py b/main.py
index 816751e..186f1a3 100644
--- a/main.py
+++ b/main.py
@@ -1,2 +1,3 @@
 print("привет, это первый коммит с тегом \"версия 1.0\"")
 print("а это уже второй коммит без тега")
+print("а этот коммит уже будет с тегом версии \"1.1\"")
\ No newline at end of file
```

Рисунок 16– результат команды git show fd50894637

Результаты команд для просмотра содержимого коммитов, и команды `git checkout -- <имя_файла>` и `git reset --hard HEAD` для отката к заданной версии, показаны на рисунках.

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git checkout -- main.py
```

Рисунок 17– результат команды `git checkout -- <имя_файла>`

```
C:\Users\FonK\Desktop\ОПИ\labRab_2>git add .
C:\Users\FonK\Desktop\ОПИ\labRab_2>git commit -m "deleted all"
[main e33827b] deleted all
 1 file changed, 6 deletions(-)
C:\Users\FonK\Desktop\ОПИ\labRab_2>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 253 bytes | 253.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/kvakaet/labRab_2.git
   ddae3af..e33827b  main -> main
C:\Users\FonK\Desktop\ОПИ\labRab_2>git reset --hard HEAD~1
HEAD is now at ddae3af commit v1.2
C:\Users\FonK\Desktop\ОПИ\labRab_2>git tag
v1.0
v1.1
v1.2
```

Рисунок 18– результат команды `git reset --hard HEAD~1`

Ответы на вопросы:

1. Как выполнить историю коммитов в Git? Какие существуют дополнительные опции для просмотра истории коммитов?

Одним из основных и наиболее мощных инструментов для этого является команда `git log`.

Опции:

Одним из самых полезных аргументов является `-p` или `--patch`, который показывает разницу (выводит патч), внесенную в каждый коммит. Так же вы можете ограничить количество записей в выводе команды;

Так же есть возможность использовать серию опций для обобщения. Например, если вы хотите увидеть сокращенную статистику для каждого коммита, вы можете использовать опцию `--stat`.

Следующей действительно полезной опцией является `--pretty`. Эта опция меняет формат вывода.

Существует несколько встроенных вариантов отображения. Опция `oneline` выводит каждый коммит в одну строку, что может быть очень удобным если вы просматриваете большое количество коммитов.

К тому же, опции `short`, `full` и `fuller` делают вывод приблизительно в том же формате, но с меньшим или большим количеством информации соответственно.

Наиболее интересной опцией является `format`, которая позволяет указать формат для вывода информации.

Полезные опции для `git log --pretty=format` отображает наиболее полезные опции для изменения формата.

Опции `oneline` и `format` являются особенно полезными с опцией `--graph` команды `log`. С этой опцией вы сможете увидеть небольшой граф в формате ASCII, который показывает текущую ветку и историю слияний.

2. Как ограничить вывод при просмотре истории коммитов?

| Опция | Описание |
|--------------------------------|---|
| <code>-(n)</code> | Показывает только последние <code>n</code> коммитов. |
| <code>--since, --after</code> | Показывает только те коммиты, которые были сделаны после указанной даты. |
| <code>--until, --before</code> | Показывает только те коммиты, которые были сделаны до указанной даты. |
| <code>--author</code> | Показывает только те коммиты, в которых запись <code>author</code> совпадает с указанной строкой. |
| <code>--committer</code> | Показывает только те коммиты, в которых запись <code>committer</code> совпадает с указанной строкой. |
| <code>--grep</code> | Показывает только коммиты, сообщение которых содержит указанную строку. |
| <code>-S</code> | Показывает только коммиты, в которых изменение в коде повлекло за собой добавление или удаление указанной строки. |

3. Как внести изменения в уже сделанный коммит?

Если вы хотите переделать коммит - внесите необходимые изменения, добавьте их в индекс и сделайте коммит ещё раз, указав параметр `--amend`: Эта команда использует область подготовки (индекс) для внесения правок в коммит.

```
$ git commit --amend
```

4. Как отменить индексацию файла в Git?

Используйте `git reset HEAD ...` для исключения из индекса.

5. Как отменить изменения в файле?

`Git checkout --` опасная команда. Все локальные изменения в файле пропадут — Git просто заменит его версией из последнего коммита.

6. Что такое удаленный репозиторий Git?

Удалённые репозитории представляют собой версии вашего проекта, сохранённые в интернете или ещё где-то в сети.

7. Как выполнить просмотр удаленных репозиториях данного локального репозитория?

Для того, чтобы просмотреть список настроенных удалённых репозиториях, вы можете запустить команду `git remote`. Она выведет названия доступных удалённых репозиториях.

8. Как добавить удаленный репозиторий для данного локального репозитория?

Для того, чтобы добавить удалённый репозиторий и присвоить ему имя (short name), просто выполните команду `git remote add`

9. Как выполнить отправку/получение изменений с удаленного репозитория?

Когда вы хотите поделиться своими наработками, вам необходимо отправить их в удалённый репозиторий. Команда для этого действия простая: `git push`

10. Как выполнить просмотр удаленного репозитория?

Если хотите получить побольше информации об одном из удалённых репозиториях, вы можете использовать команду `git remote show<remote>`.

11. Каково назначение тэгов Git?

Как и большинство СКВ, Git имеет возможность пометить определённые моменты в истории как важные. Такие пометки в Git называются тегами.

12. Как осуществляется работа с тэгами Git?

Просмотреть список имеющихся тегов в Git можно очень просто. Достаточно набрать команду `git tag` (параметры `-l` и `--list` опциональны): Данная команда перечисляет теги в алфавитном порядке; порядок их отображения не имеет существенного значения. Так же можно выполнить поиск тега по шаблону.

Создание тегов Git использует два основных типа тегов: легковесные и аннотированные. Легковесный тег — это что-то очень похожее на ветку, которая не изменяется — просто указатель на определённый коммит. А вот аннотированные теги хранятся в базе данных Git как полноценные объекты. Обычно рекомендуется создавать аннотированные теги, чтобы иметь всю перечисленную информацию; но если вы хотите сделать временную метку или по какой-то причине не хотите сохранять остальную информацию, то для этого годятся и легковесные.

Создание аннотированного тега в Git.

Самый простой способ — это указать - а при выполнении команды `tag`: Опция `-m` задаёт сообщение, которое будет храниться вместе с тегом. Если не указать сообщение, то Git запустит редактор, чтобы вы смогли его ввести. С помощью команды `git show` вы можете посмотреть данные тега вместе с коммитом.

Обмен тегами. Процесс аналогичен отправке веток — достаточно выполнить команду `git push origin <tag name>`. Если у вас много тегов, и вам хотелось бы отправить все за один раз, то можно использовать опцию `--tags` для команды `git push`. В таком случае все ваши теги отправятся на удалённый сервер (если только их уже там нет).

Удаление тегов.

Для удаления тега в локальной репозитории достаточно выполнить команду `git tag -d`.

Переход на тег.

Если вы хотите получить версии файлов, на которые указывает тег, то вы можете сделать `git checkout` для тега.

13. Самостоятельно изучите назначение флага `--prune` в командах `git fetch` и `git push`. Каково назначение этого флага?

`--prune`

Если вы хотите получить все данные и одновременно удалить удаленные данные, добавьте флаг `--prune` в команде `git fetch`

Удалите удаленные ветки, у которых нет локального аналога. Например, удаленная ветка tmp будет удалена, если локальная ветка с таким же именем больше не существует. Это также учитывает refspecs, например `git push -- prune remote refs/heads/*:refs/tmp/*`, убедитесь, что

удаленный объект refs/tmp/foобудет удален, если refs/heads/foo он не существует.