

Git и GitHub

Системы контроля версий

- SVN
- Git
- ~~Mercurial~~

Git

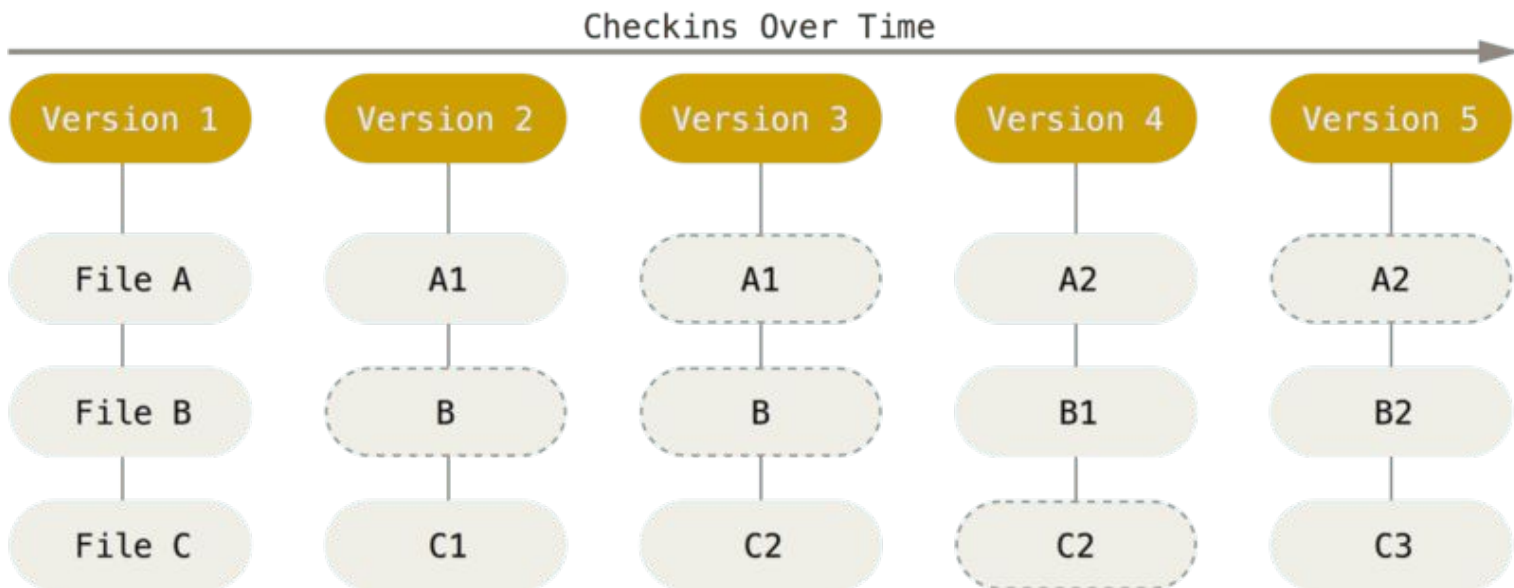
Консольная утилита - <https://git-scm.com/>

Десктопные клиенты:

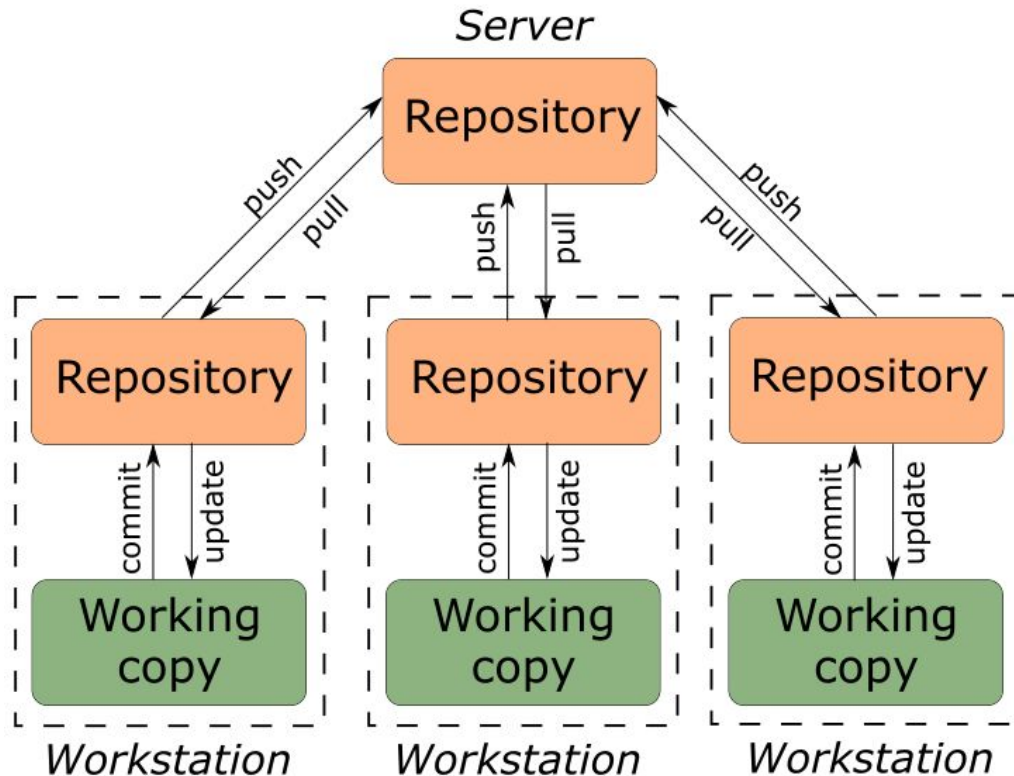
GitHub Desktop - <https://desktop.github.com/>

SourceTree - <https://www.sourcetreeapp.com/>

Хранение версий в GIT



GIT - распределенная система



Создание репозитория

Пустой локальный репозиторий

```
cd /my_project  
git init
```

Клонирование удаленного репозитория

```
cd /some_folder  
git clone https://github.com/path_to/my_project.git
```

Работа с git

Создадим файл <name>.py

```
git add *.py # добавляет файл в индекс
```

```
git commit -m "initial commit"
```

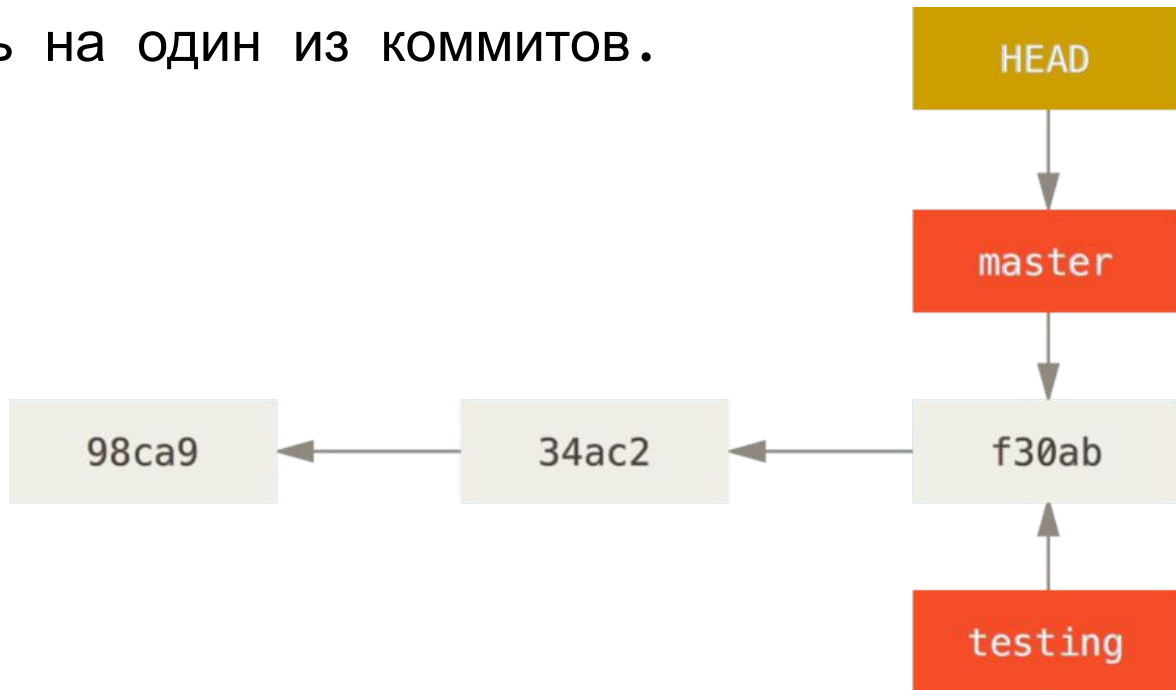
```
git status
```

```
git log
```

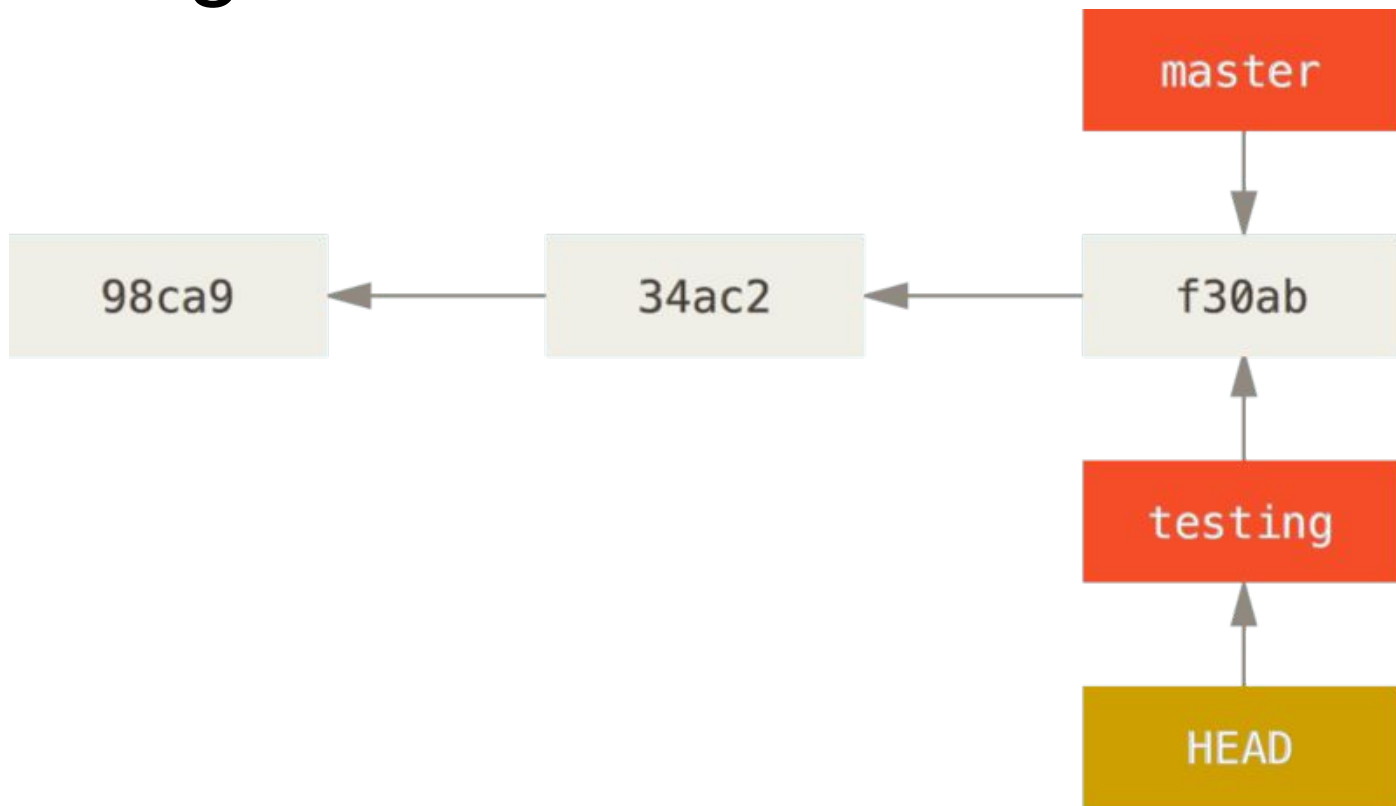
```
git push # отправляет изменения во внешний репозиторий
```

Ветки в git

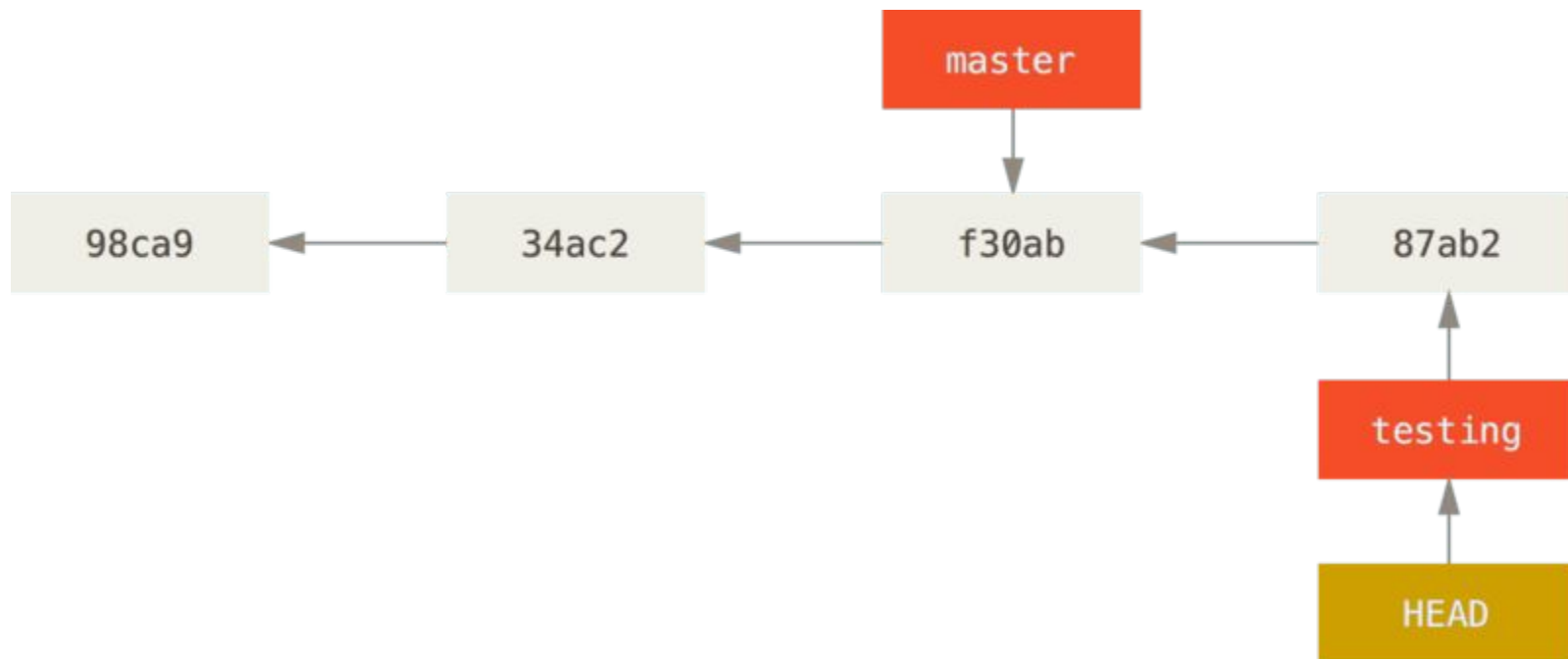
Указатель на один из КОММИТОВ.



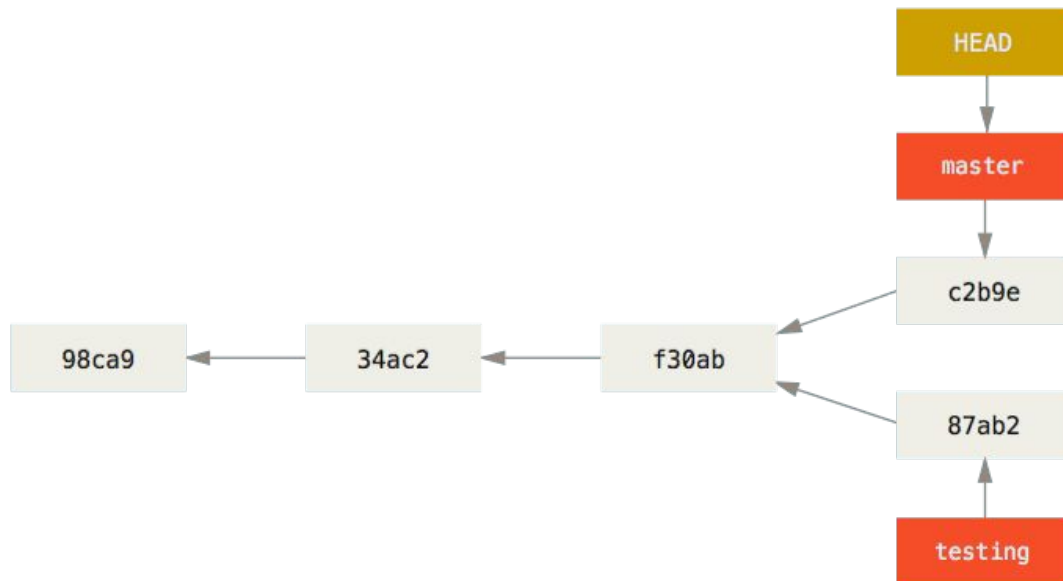
Ветки в git



Ветки в git



Ветки в git



Ветки в git

`git status` # информация о текущем статусе

`git branch` # список веток

`git branch lecture_1` # создание новой ветки с именем `lecture_1`

`git checkout lecture_1` # переключаемся на созданную ветку

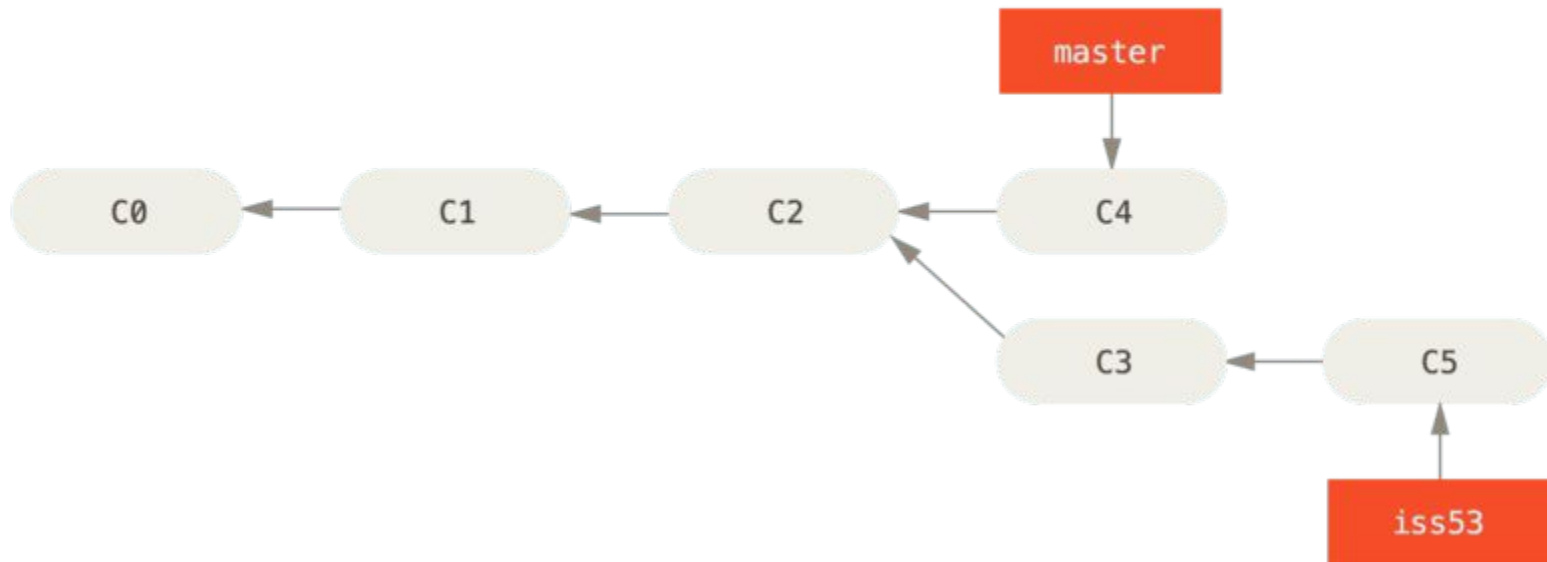
`git checkout -b lecture_1` # создаем ветку и переключаемся на нее

`git push -u origin lecture_1` # отправляем изменения ветки `lecture_1` во внешний репозиторий

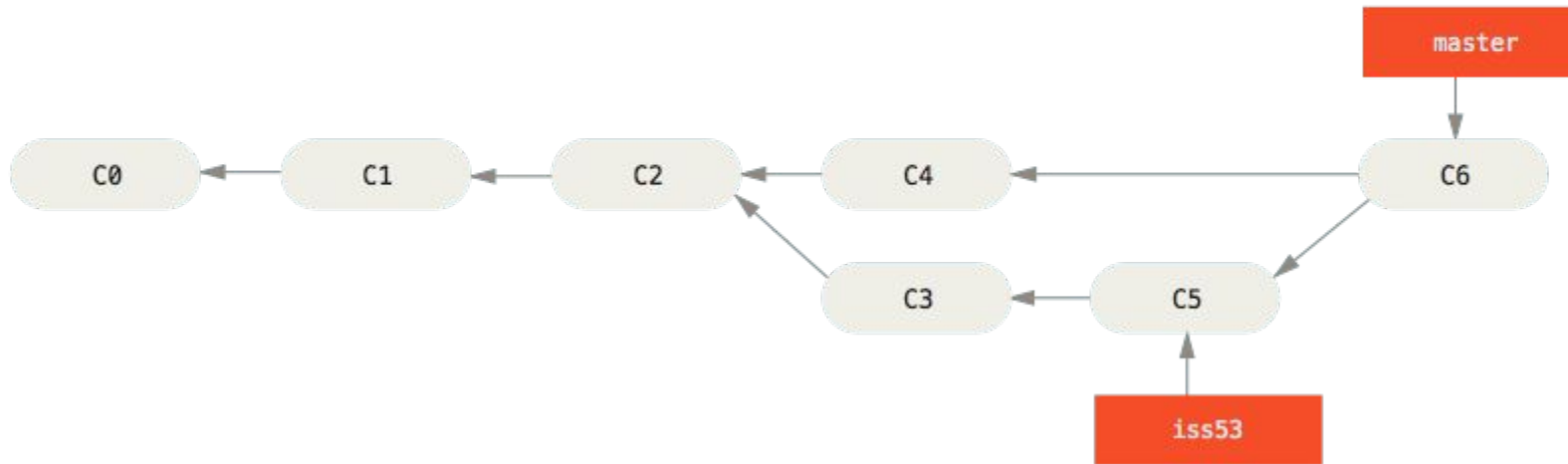
`git checkout main` # переключение обратно на ветку `main`

`git pull` # получение свежей версии с сервера

Слияние веток (merge)



Слияние веток (merge)

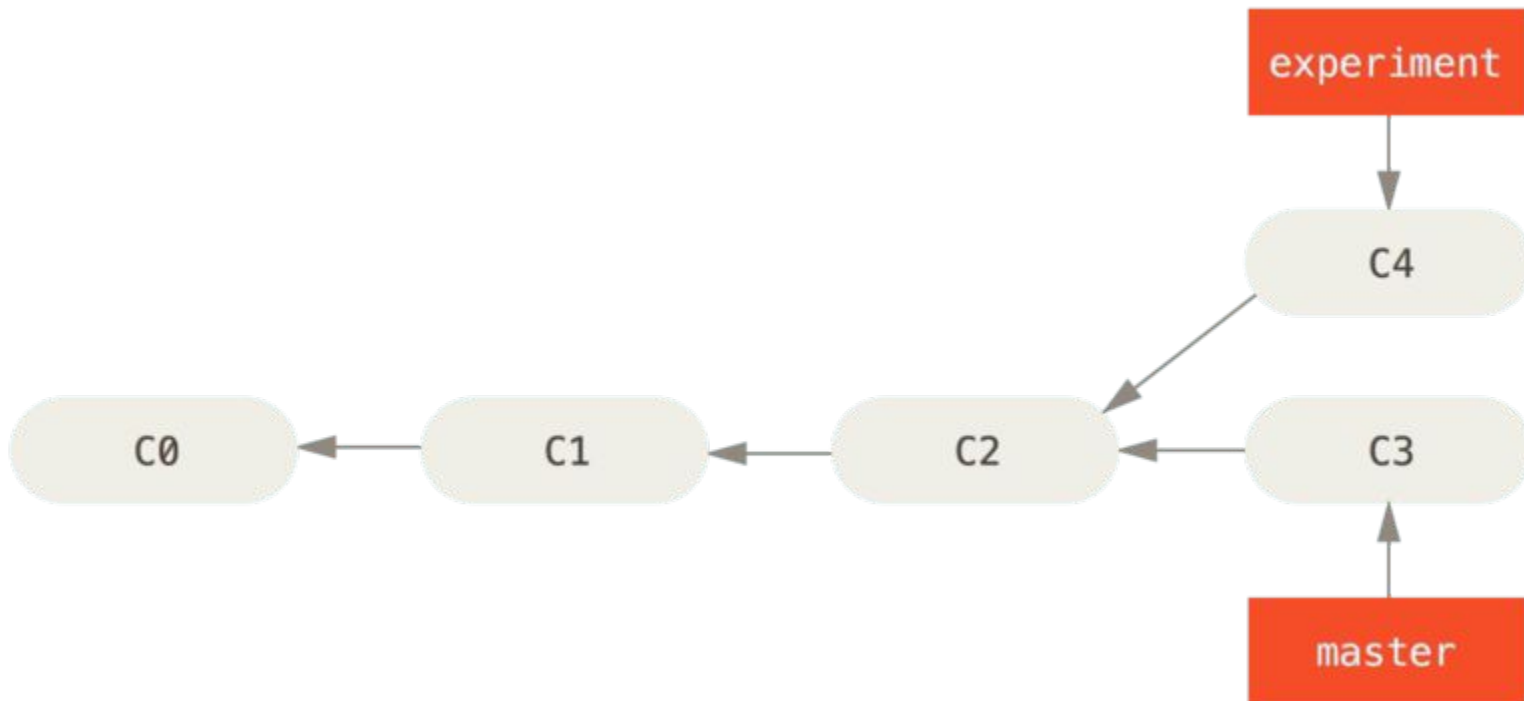


Слияние веток (merge)

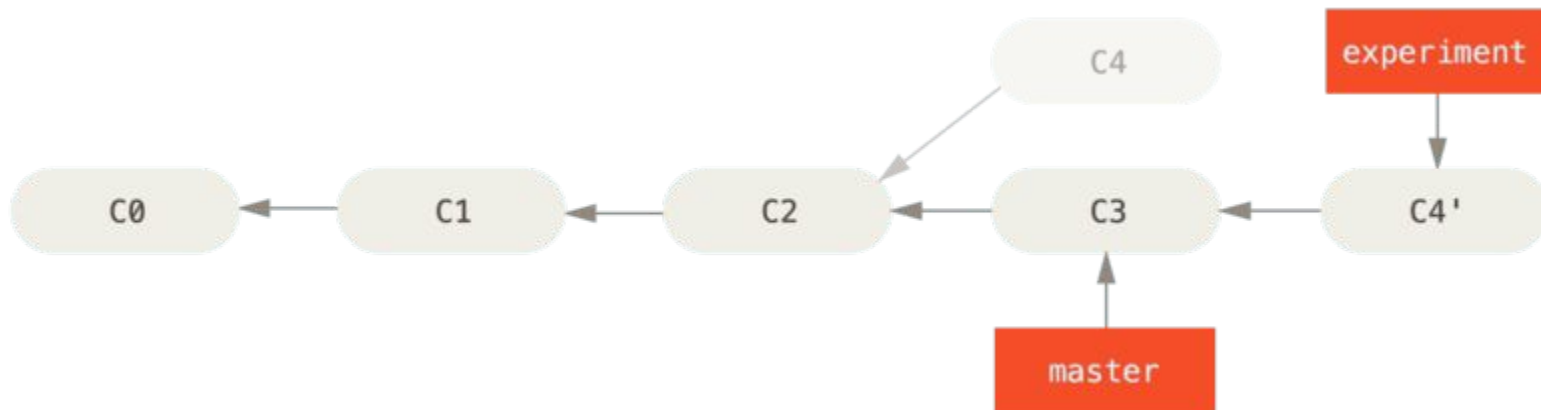
```
git checkout main
```

```
git merge iss53
```

Перебазирование веток (rebase)



Перебазирование веток (rebase)



Перебазирование веток (rebase)

```
git checkout experiment  
git rebase main
```

Исправление конфликтов:

```
git rebase --continue
```

.gitignore

Скрытие файлов и папок от системы контроля версий.

Пример файла:

```
build/  
.ipynb_checkpoints  
*.log  
# Environments  
env/  
venv/  
.idea
```

Работа с GitHub

Кнопка “New pull request” в интерфейсе на вкладке “Pull requests”

Выбор веток:

base - main




compare - <ваша ветка>


“Merge pull request” - только после получения Approve


Работа с GitHub

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).








 base: main  compare: testing  **Able to merge.** These branches can be automatically merged.




Название пул-реквеста 

Write


Preview

H B I  <>     @  

Описание изменений, которые содержатся в пул-реквесте.

Attach files by dragging & dropping, selecting or pasting them. 

Create pull request

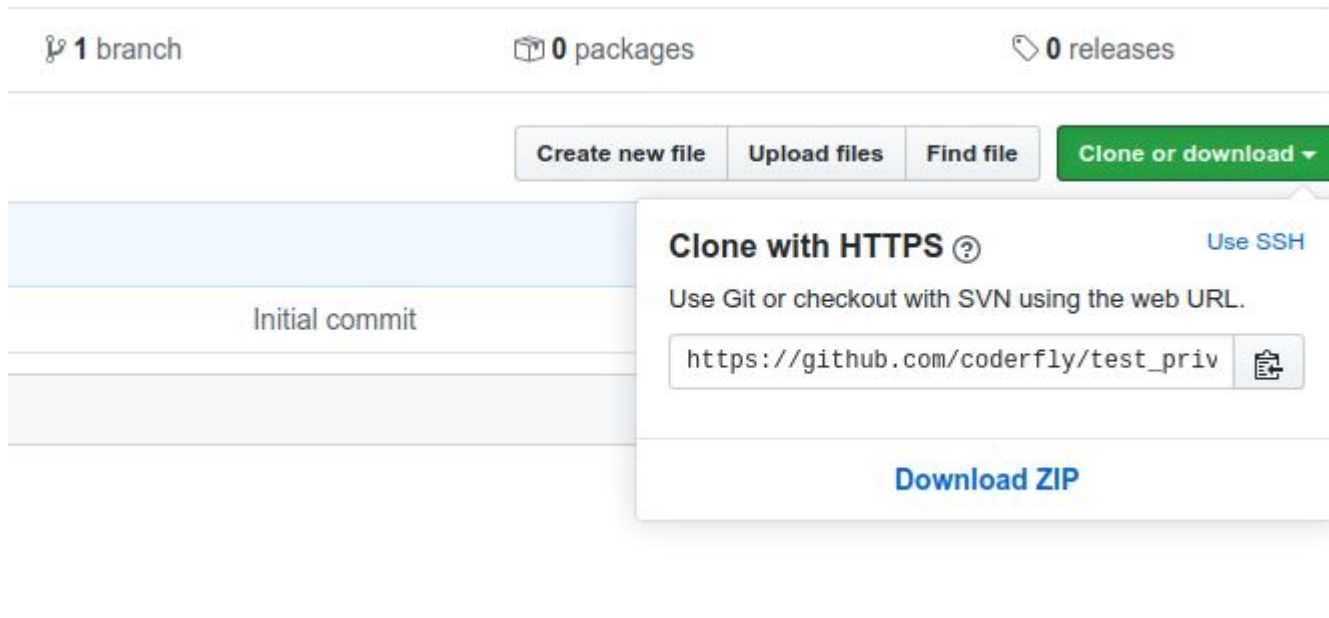
 Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Ревью кода

- архитектура
- соответствие требованиям поставленной задачи
- нейминг
- возможные баги/проблемные места
- покрытие тестами
- соответствие стиля

Упражнение 1

Склонируйте созданный на GitHub репозиторий.



Упражнение 1

Создайте в корне файл test.py с содержимым:

```
print("First branch")
```

Добавьте файл в индекс, сделайте коммит, выполните push во внешний репозиторий.

Упражнение 2

1. Создайте ветку `issue-1`, переключитесь на нее.
2. Создайте папку `example`
3. В созданной папке создайте файл `task1.py` с содержимым:

```
print("Second branch")
```
4. Добавьте файл в индекс, сделайте коммит, выполните `push` во внешний репозиторий.

Упражнение 2

5. Переключитесь на ветку `main`, и проверьте что там нет вашей папки и файла.
6. Создайте пулл-реквест на гитхабе.
7. Вмержите пулл-реквест на гитхабе.
8. Обновите локальное содержимое ветки `main`, убедитесь что в ней добавились новые папка и файл.