Технологии разработки программного обеспечения

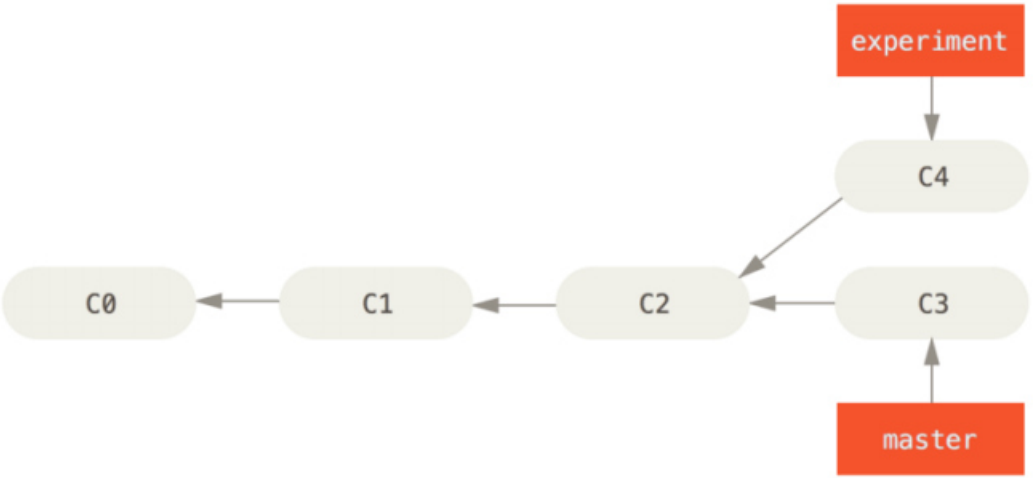
Старший преподаватель Кафедры вычислительных систем
Елизавета Ивановна Токмàшева

email: eliz_tokmasheva@sibguti.ru

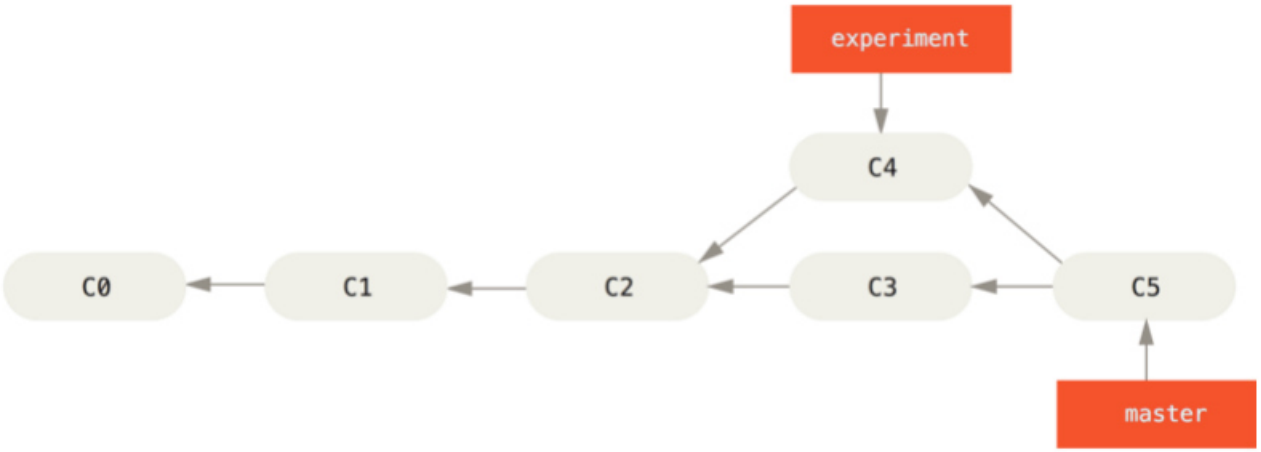
2024, 1 курс, 2 семестр

Ветвление в Git
Rebase

История до слияния:



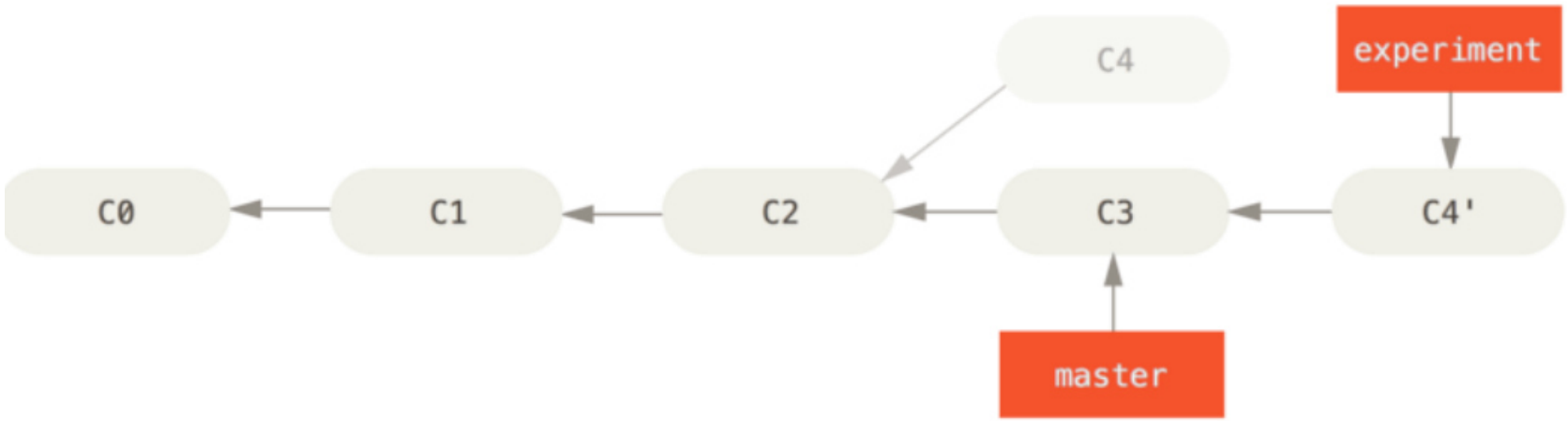
После слияния:



Ветвление в Git
Rebase

Применение изменений из коммита C4 поверх коммита C3, называется *перебазированием* (rebase)

$ git switch experiment
$ git rebase master

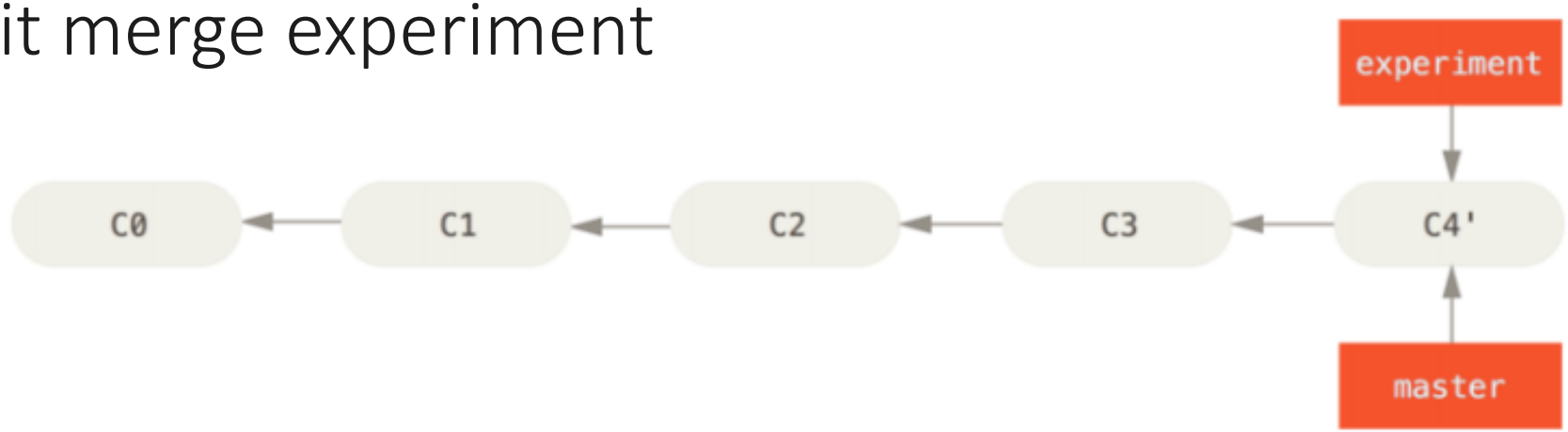


Ветвление в Git
Rebase

Далее можно переключиться на ветку master и обновить её с помощью слияния с перемоткой

$ git switch master

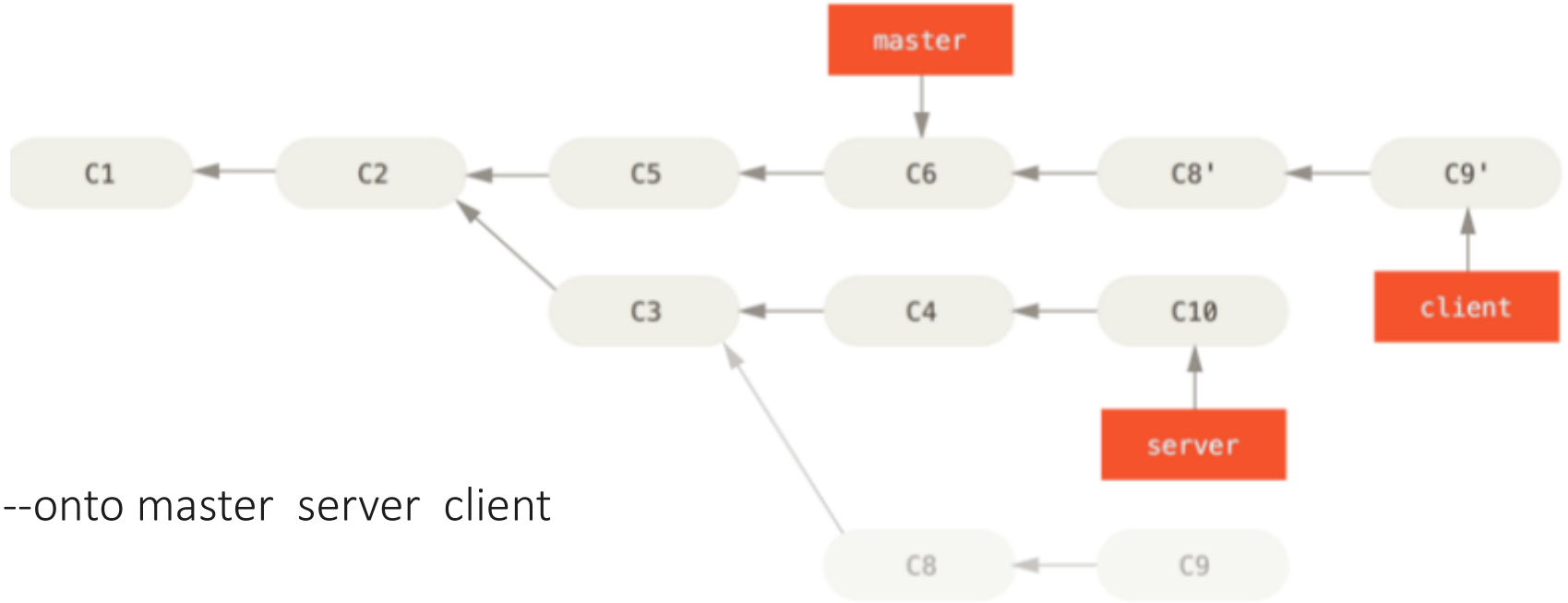
$ git merge experiment



Ветвление в Git
Rebase. Расширенные возможности

Опция--onto позволяет передавать конкретные ссылки в качестве основания для перебазирования

$ git rebase--onto <newbase> [<upstream> [<branch>]]

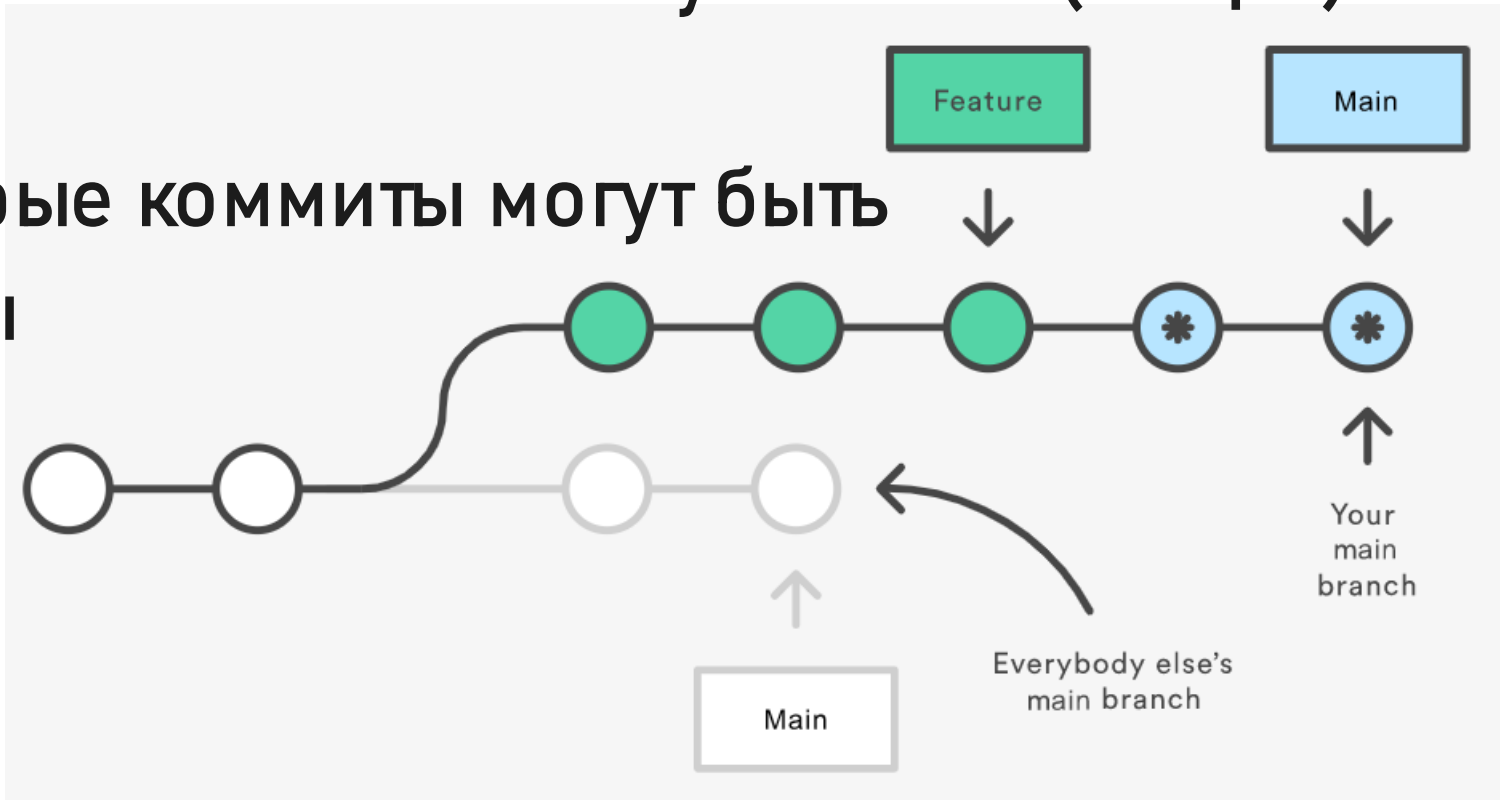


$ git rebase --onto master server client

Ветвление в Git
Rebase

`git rebase`

- выравнивает историю изменений
- нельзя использовать на публичных (общих) ветках
- некоторые коммиты могут быть утрачены



Ветвление в Git
Работа с удаленными репозиториями

Посмотреть список настроенных удаленных репозиториев

```
$ git remote
```

Если репозиторий был склонирован, то команда выведет имя по умолчанию удаленного репозитория — origin

Ветвление в Git
Работа с удаленными репозиториями

```
eliz@LAPTOP-F6OV9FH5:~$ git clone https://github.com/open-mpi/ompi
Cloning into 'ompi'...
remote: Enumerating objects: 462241, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 462241 (delta 12), reused 6 (delta 3), pack-reused 462217
Receiving objects: 100% (462241/462241), 156.03 MiB | 3.21 MiB/s, done.
Resolving deltas: 100% (375932/375932), done.
Updating files: 100% (5970/5970), done.
eliz@LAPTOP-F6OV9FH5:~$ cd ompi/
eliz@LAPTOP-F6OV9FH5:~/ompi$ git remote
origin
eliz@LAPTOP-F6OV9FH5:~/ompi$ git remote -v
origin  https://github.com/open-mpi/ompi (fetch)
origin  https://github.com/open-mpi/ompi (push)
```

Ветвление в Git
Работа с удаленными репозиториями

Чтобы добавить удаленный репозиторий и присвоить ему имя (shortname) выполните команду

$ git remote add <name> <url>

```
eliz@LAPTOP-F6OV9FH5:~/ompi$ git remote add ompi-test https://github.com/open-mpi/ompi-test
eliz@LAPTOP-F6OV9FH5:~/ompi$ git remote -v
ompi-test       https://github.com/open-mpi/ompi-test (fetch)
ompi-test       https://github.com/open-mpi/ompi-test (push)
origin  https://github.com/open-mpi/ompi (fetch)
origin  https://github.com/open-mpi/ompi (push)
```

Ветвление в Git
Работа с удаленными репозиториями

Для получения данных из удаленных проектов используется команда

```
$ git fetch [remote-name]
```

или

```
$ git pull [remote-name]
```

git pull = git fetch + git merge

Для отправки изменений в удаленный репозиторий

```
$ git push [remote-name] [branch-name]
```

Ветвление в Git
Удаленные ветки

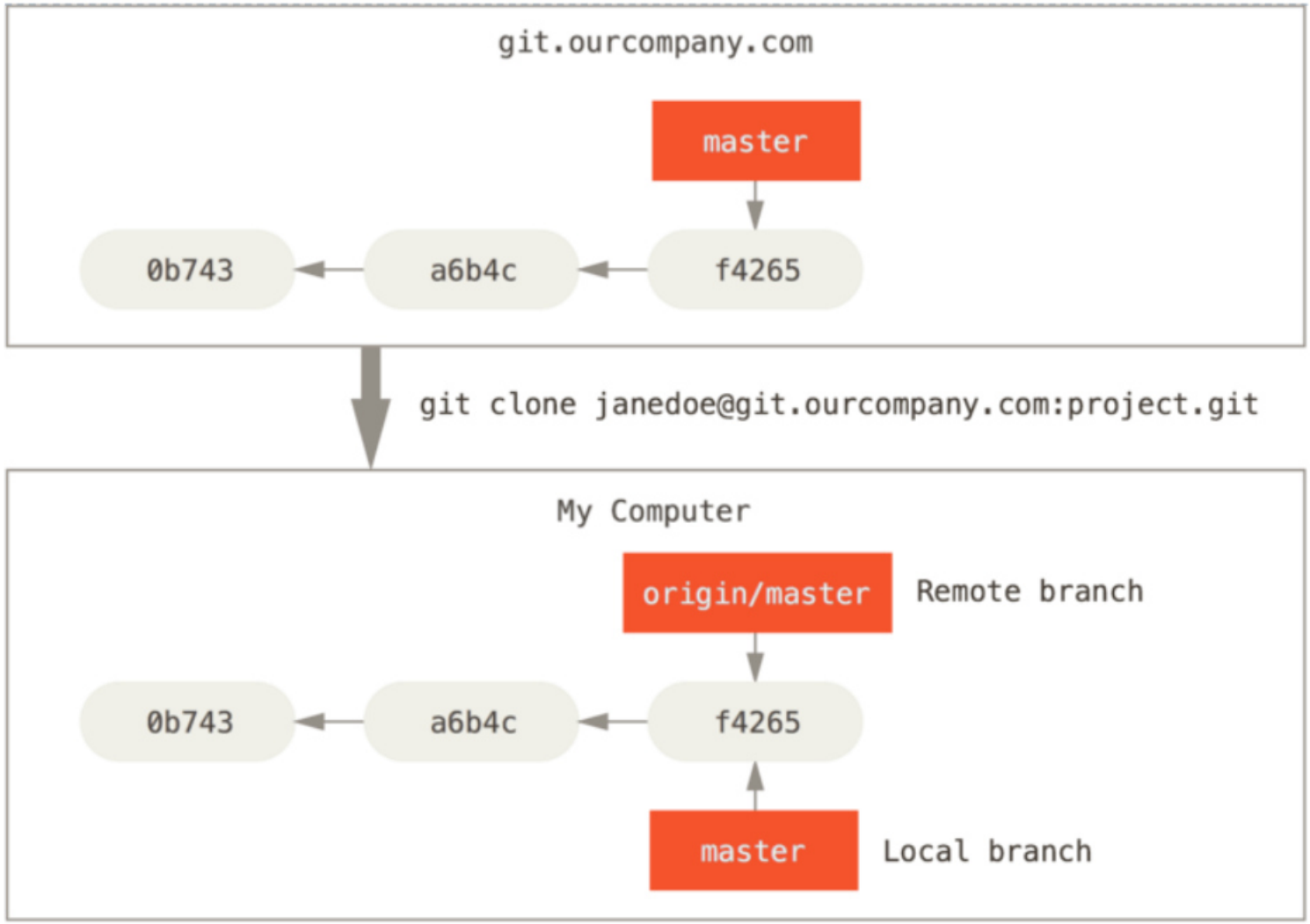
Удаленные ветки — это ссылки на состояние веток в удаленных репозиториях. Являются локальными ветками, которые нельзя перемещать. Они перемещаются всякий раз, когда пользователь отправляет изменения на сервер (git push).

Вид удаленной ветки

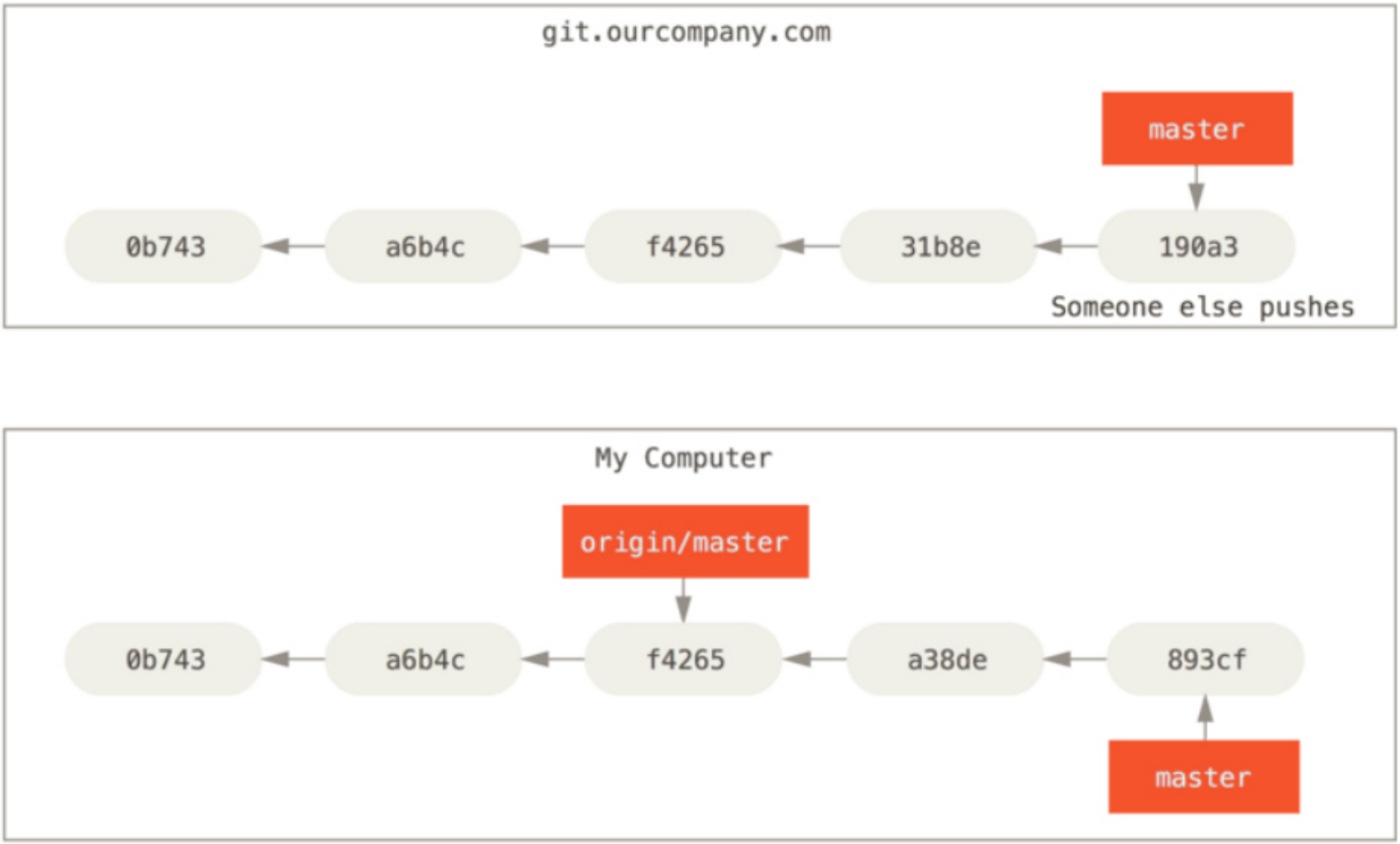
<remote-name>/<branch-name>

Например, origin/main

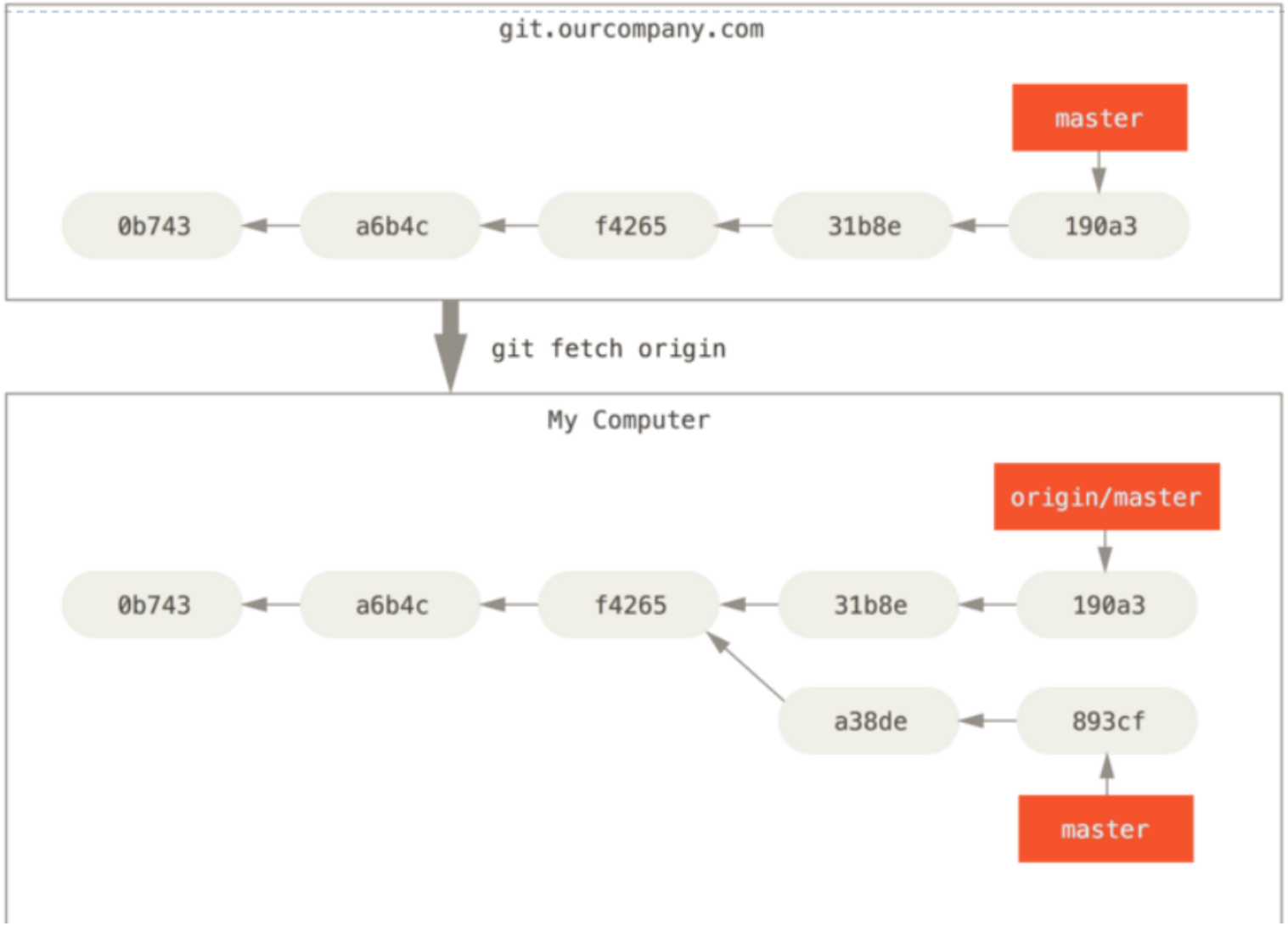
Ветвление в Git
Удаленные ветки



Ветвление в Git
Удаленные ветки



Ветвление в Git
Удаленные ветки



Данные полученные в результате выполнения команды git fetch попадают в staged-область

Ветвление в Git
push/fetch

Локальные ветки автоматически не синхронизируются с удаленными серверами — нужно явно отправлять те ветки, которыми хотите поделиться

```
$ git push origin feature
```

Чтобы получить изменения из ветки feature, выполните

```
$ git fetch origin

From https://github.com/schacon/simplegit
 * [new branch] feature-> origin/feature
```

В этом случае вы получите только ссылку на удаленную ветку — указатель origin/feature, который нельзя изменить

Ветвление в Git
Отслеживание веток

Отслеживаемые ветки — это локальные ветки, которые напрямую связаны с удаленной веткой

```
$ git switch-b [branch] [remote-name]/[branch]
```

```
$ git switch --track origin/develop

Branch develop set up to track remote branch develop
from origin.
Switched to a new branch 'develop'
```

Ветвление в Git
Конфликты

Конфликты возникают, когда одна и та же часть одного и того же файла была изменена по-разному в двух объединяемых ветках

В этом случае Git не может автоматически объединить ветки

```
eliz@LAPTOP-F6OV9FH5:~/5-merge-conflict-easy$ git merge origin/develop
Auto-merging main.c
CONFLICT (content): Merge conflict in main.c
Automatic merge failed; fix conflicts and then commit the result.
```

Ветвление в Git
Конфликты

Git приостанавливает процесс слияния до тех пор, пока вы самостоятельно не разрешите конфликт

```
eliz@LAPTOP-F6OV9FH5:~/5-merge-conflict-easy$ git status
On branch main
Your branch is up to date with 'origin/main'.

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   main.c

no changes added to commit (use "git add" and/or "git commit -a")
```

Ветвление в Git
Конфликты

```
int main()
{
    int arr[] = {3, 1, 4, 1, 5, 9, 2, 6, 5, 3};

<<<<<<< HEAD
    const int min = min_element(arr, 10);

    printf("Array: ");
    size_t i;
    for (i = 0; i < 10; ++i) {
        printf("%d ", arr[i]);
=======
    const int min = min_element(arr, N_ELEMENTS(arr));

    printf("Array: ");
    size_t i = 0;
    printf("%d", arr[i]);
    ++i;
    for (; i < N_ELEMENTS(arr); ++i) {
        printf(", %d", arr[i]);
>>>>>>> origin/develop
    }
    printf("\n");

    printf("Min element: %d\n", min);

    return 0;
}
```

Для разрешения конфликта оставьте нужную часть кода и отметьте разрешение конфликта, выполнив команду

$ git add -u

Ветвление в Git
Конфликты

- Если конфликт возник на этапе выполнения слияния, завершение разрешение конфликта осуществляется созданием коммита слияния

  $ git commit

- Если конфликт возник при выполнении ребейза, то продолжить перебазирование

  $ git rebase --continue

  Отмена ребейза:

  $ git rebase --abort

  Пропуск коммита для перемещения:

  $ git rebase --skip