3.5 Ветвление в Git - Удалённые ветки

**Удалённые ветки**

Удалённые ссылки — это ссылки (указатели) в ваших удалённых репозиториях, включая ветки, теги и так далее. Полный список удалённых ссылок можно получить с помощью команды git ls-remote <remote> или команды git remote show <remote> для получения удалённых веток и дополнительной информации. Тем не менее, более распространенным способом является использование веток слежения.

Ветки слежения — это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые нельзя перемещать; Git перемещает их автоматически при любой коммуникации с удаленным репозиторием, чтобы гарантировать точное соответствие с ним. Представляйте их как закладки для напоминания о том, где ветки в удалённых репозиториях находились во время последнего подключения к ним.

Имена веток слежения имеют вид <remote>/<branch>. Например, если вы хотите посмотреть, как выглядела ветка master на сервере origin во время последнего соединения с ним, используйте ветку origin/master. Если вы с коллегой работали над одной задачей и он отправил на сервер ветку iss53, при том что у вас может быть своя локальная ветка iss53, удалённая ветка будет представлена веткой слежения с именем origin/iss53.

Возможно, всё это сбивает с толку, поэтому давайте рассмотрим на примере. Скажем, у вас в сети есть свой Git-сервер с адресом git.ourcompany.com. Если вы с него что-то склонируете, команда clone автоматически назовёт его origin, заберёт оттуда все данные, создаст указатель на то, на что там указывает ветка master, и назовёт его локально origin/master. Git также создаст вам локальную ветку master, которая будет начинаться там же, где и ветка master в origin, так что вам будет с чего начать.

|  |  |
| --- | --- |
| **Note** | **“origin” — это не специальное название**  Подобно названию ветки “master”, “origin” не имеет какого-либо специального значения в Git. В то время как “master” — это название по умолчанию для ветки при выполнении git init только потому, что часто используется, “origin” — это название по умолчанию для удалённого сервера, когда вы запускаете git clone. Если вы выполните git clone -o booyah, то по умолчанию ветка слежения будет иметь вид booyah/master. |



Рисунок 30. Серверный и локальный репозитории после клонирования

Если вы сделаете что-то в своей локальной ветке master, а тем временем кто-то отправит изменения на сервер git.ourcompany.com и обновит там ветку master, то ваши истории продолжатся по-разному. Пока вы не свяжетесь с сервером origin ваш указатель origin/master останется на месте.



Рисунок 31. Локальная и удалённая работа может расходиться

Для синхронизации ваших изменений с удаленным сервером выполните команду git fetch <remote> (в нашем случае git fetch origin). Эта команда определяет какому серверу соответствует “origin” (в нашем случае это git.ourcompany.com), извлекает оттуда данные, которых у вас ещё нет, и обновляет локальную базу данных, сдвигая указатель origin/master на новую позицию.



Рисунок 32. git fetch обновляет ветки слежения

Чтобы продемонстрировать, как будут выглядеть удалённые ветки в ситуации с несколькими удалёнными серверами, предположим, что у вас есть ещё один внутренний Git-сервер, который используется для разработки только одной из ваших команд разработчиков. Этот сервер находится на git.team1.ourcompany.com. Вы можете добавить его в качестве новой удалённой ссылки для текущего проекта с помощью команды git remote add, как было описано в [[ch02-git-basics]](https://git-scm.com/book/ru/v2/ch00/ch02-git-basics). Назовите этот удалённый сервер teamone — это имя будет сокращением вместо полного URL.



Рисунок 33. Добавление ещё одного сервера в качестве удалённой ветки

Теперь вы можете выполнить команду git fetch teamone для получения всех изменений с сервера teamone, которых у вас нет локально. Так как в данный момент на этом сервере есть только те данные, что содержит сервер origin, Git ничего не получит, но создаст ветку слежения с именем teamone/master, которая будет указывать на тот же коммит, что и ветка master на сервере teamone.



Рисунок 34. Ветка слежения teamone/master

**Отправка изменений**

Когда вы хотите поделиться веткой, вам необходимо отправить её на удалённый сервер, где у вас есть права на запись. Ваши локальные ветки автоматически не синхронизируются с удалёнными при отправке — вам нужно явно указать те ветки, которые вы хотите отправить. Таким образом, вы можете использовать свои личные ветки для работы, которую не хотите показывать, а отправлять только те тематические ветки, над которыми вы хотите работать с кем-то совместно.

Если у вас есть ветка serverfix, над которой вы хотите работать с кем-то ещё, вы можете отправить её точно так же, как вы отправляли вашу первую ветку. Выполните команду git push <remote> <branch>:

$ git push origin serverfix

Counting objects: 24, done.

Delta compression using up to 8 threads.

Compressing objects: 100% (15/15), done.

Writing objects: 100% (24/24), 1.91 KiB | 0 bytes/s, done.

Total 24 (delta 2), reused 0 (delta 0)

To https://github.com/schacon/simplegit

\* [new branch] serverfix -> serverfix

Это в некотором роде сокращение. Git автоматически разворачивает имя ветки serverfix до refs/heads/serverfix:refs/heads/serverfix, что означает “возьми мою локальную ветку serverfix и обнови ей удалённую ветку serverfix”. Мы подробно рассмотрим часть с refs/heads/ в [[ch10-git-internals]](https://git-scm.com/book/ru/v2/ch00/ch10-git-internals), но обычно её можно пропустить. Вы также можете выполнить git push origin serverfix:serverfix — произойдёт то же самое — здесь говорится “возьми мою ветку serverfix и сделай её удалённой веткой serverfix”. Можно использовать этот формат для отправки локальной ветки в удалённую ветку с другим именем. Если вы не хотите, чтобы на удалённом сервере ветка называлась serverfix, то вместо предыдущей команды выполните git push origin serverfix:awesomebranch, которая отправит локальную ветку serverfix в ветку awesomebranch удалённого репозитория.

|  |  |
| --- | --- |
| **Note** | **Не вводите каждый раз свой пароль**  Если вы используете HTTPS URL для отправки изменений, Git-сервер будет спрашивать имя пользователя и пароль для аутентификации. По умолчанию вам будет предложено ввести эти данные в терминале, чтобы сервер мог определить разрешена ли вам отправка изменений.  Если вы не хотите вводить свои данные каждый раз при отправке изменений, вы можете настроить “credential cache”. Проще всего держать их в памяти несколько минут, это легко настроить с помощью команды git config --global credential.helper cache.  Для получения более подробной информации о различных вариантах кэша учётных данных обратитесь к разделу [Хранилище учётных данных](https://git-scm.com/book/ru/v2/ch00/r_credential_caching). |

В следующий раз, когда один из ваших соавторов будет получать обновления с сервера, он получит ссылку на то, на что указывает serverfix на сервере, как удалённую ветку origin/serverfix:

$ git fetch origin

remote: Counting objects: 7, done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 3 (delta 0)

Unpacking objects: 100% (3/3), done.

From https://github.com/schacon/simplegit

\* [new branch] serverfix -> origin/serverfix

Необходимо отметить, что при получении данных создаются ветки слежения, вы не получаете автоматически для них локальных редактируемых копий. Другими словами, в нашем случае вы не получите новую ветку serverfix — только указатель origin/serverfix, который вы не можете изменять.

Чтобы слить эти наработки в свою текущую рабочую ветку, выполните git merge origin/serverfix. Если вам нужна локальная ветка serverfix, в которой вы сможете работать, то вы можете создать её на основе ветки слежения:

$ git checkout -b serverfix origin/serverfix

Branch serverfix set up to track remote branch serverfix from origin.

Switched to a new branch 'serverfix'

Это даст вам локальную ветку, в которой можно работать и которая будет начинаться там же, где и origin/serverfix.

**Отслеживание веток**

Получение локальной ветки из удалённой ветки автоматически создаёт то, что называется “веткой слежения” (а ветка, за которой следит локальная называется “upstream branch”). Ветки слежения — это локальные ветки, которые напрямую связаны с удалённой веткой. Если, находясь на ветке слежения, выполнить git pull, то Git уже будет знать с какого сервера получать данные и какую ветку использовать для слияния.

При клонировании репозитория, как правило, автоматически создаётся ветка master, которая следит за origin/master. Однако, при желании вы можете настроить отслеживание и других веток — следить за ветками на других серверах или отключить слежение за веткой master. Вы только что видели простейший пример, что сделать это можно с помощью команды git checkout -b <branch> <remote>/<branch>. Это часто используемая команда, поэтому Git предоставляет сокращённую форму записи в виде флага --track:

$ git checkout --track origin/serverfix

Branch serverfix set up to track remote branch serverfix from origin.

Switched to a new branch 'serverfix'

В действительности, это настолько распространённая команда, что существует сокращение для этого сокращения. Если вы пытаетесь извлечь ветку, которая не существует, но существует только одна удалённая ветка с точно таким же именем, то Git автоматически создаст ветку слежения:

$ git checkout serverfix

Branch serverfix set up to track remote branch serverfix from origin.

Switched to a new branch 'serverfix'

Чтобы создать локальную ветку с именем, отличным от имени удалённой ветки, просто укажите другое имя:

$ git checkout -b sf origin/serverfix

Branch sf set up to track remote branch serverfix from origin.

Switched to a new branch 'sf'

Теперь ваша локальная ветка sf будет автоматически получать изменения из origin/serverfix.

Если у вас уже есть локальная ветка и вы хотите настроить ее на слежение за удалённой веткой, которую вы только что получили, или хотите изменить используемую upstream-ветку, то воспользуйтесь параметрами -u или --set-upstream-to для команды git branch, чтобы явно установить новое значение.

$ git branch -u origin/serverfix

Branch serverfix set up to track remote branch serverfix from origin.

|  |  |
| --- | --- |
| **Note** | **Сокращение Upstream**  Если у вас настроена отслеживаемая ветка, вы можете ссылаться на нее с помощью сокращений @{upstream} или @{u}. Итак, если вы находитесь на ветке master и она следит за origin/master, при желании вы можете использовать git merge @{u} вместо git merge origin/master. |

Если вы хотите посмотреть как у вас настроены ветки слежения, воспользуйтесь опцией -vv для команды git branch. Это выведет список локальных веток и дополнительную информацию о том, какая из веток отслеживается, отстаёт, опережает или всё сразу относительно отслеживаемой.

$ git branch -vv

iss53 7e424c3 [origin/iss53: ahead 2] forgot the brackets

master 1ae2a45 [origin/master] deploying index fix

\* serverfix f8674d9 [teamone/server-fix-good: ahead 3, behind 1] this should do it

testing 5ea463a trying something new

Итак, здесь мы видим, что наша ветка iss53 следит за origin/iss53 и “опережает” её на два изменения — это значит, что у нас есть два локальных коммита, которые не отправлены на сервер. Мы также видим, что наша ветка master отслеживает ветку origin/master и находится в актуальном состоянии. Далее мы можем видеть, что локальная ветка serverfix следит за веткой server-fix-good на сервере teamone, опережает её на три коммита и отстает на один — это значит, что на сервере есть один коммит, который мы ещё не слили, и три локальных коммита, которые ещё не отправлены на сервер. В конце мы видим, что наша ветка testing не отслеживает удаленную ветку.

Важно отметить, что эти цифры описывают состояние на момент последнего получения данных с каждого из серверов. Эта команда не обращается к серверам, а лишь говорит вам о том, какая информация с этих серверов сохранена в локальном кэше. Если вы хотите иметь актуальную информацию об этих числах, вам необходимо получить данные со всех ваших удалённых серверов перед запуском команды. Сделать это можно вот так:

$ git fetch --all; git branch -vv

**Получение изменений**

Команда git fetch получает с сервера все изменения, которых у вас ещё нет, но не будет изменять состояние вашей рабочей директории. Эта команда просто получает данные и позволяет вам самостоятельно сделать слияние. Тем не менее, существует команда git pull, которая в большинстве случаев является командой git fetch, за которой непосредственно следует команда git merge. Если у вас настроена ветка слежения как показано в предыдущем разделе, или она явно установлена, или она была создана автоматически командами clone или checkout, git pull определит сервер и ветку, за которыми следит ваша текущая ветка, получит данные с этого сервера и затем попытается слить удалённую ветку.

Обычно, лучше явно использовать команды fetch и merge, поскольку магия git pull может часто сбивать с толку.

**Удаление веток на удалённом сервере**

Скажем, вы и ваши соавторы закончили с нововведением и слили его в ветку master на удалённом сервере (или в какую-то другую ветку, где хранится стабильный код). Вы можете удалить ветку на удалённом сервере используя параметр --delete для команды git push. Для удаления ветки serverfix на сервере, выполните следующую команду:

$ git push origin --delete serverfix

To https://github.com/schacon/simplegit

- [deleted] serverfix

Всё, что делает эта строка — удаляет указатель на сервере. Как правило, Git сервер хранит данные пока не запустится сборщик мусора, поэтому если ветка была удалена случайно, чаще всего её легко восстановить.

[prev](https://git-scm.com/book/ru/v2/%D0%92%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D0%B2-Git-%D0%A0%D0%B0%D0%B1%D0%BE%D1%82%D0%B0-%D1%81-%D0%B2%D0%B5%D1%82%D0%BA%D0%B0%D0%BC%D0%B8) | [next](https://git-scm.com/book/ru/v2/%D0%92%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D0%B2-Git-%D0%9F%D0%B5%D1%80%D0%B5%D0%B1%D0%B0%D0%B7%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

[About this site](https://git-scm.com/site)  
Patches, suggestions, and comments are welcome.