GeekBrains

Git. Базовый курс

Урок 9

Версионирование

Версия программного обеспечения

Версии программного обеспечения, как правило, состоит из трех цифр, первая из которых называется мажорной, вторая — минорной, а третья — патч-версией.

Мажорная версия меняется довольно редко, как правило, в случае кардинальной переработки программного продукта.

Минорная версия отвечает за регулярные релизы, в рамках которых добавляется новый функционал и исправляются ошибки.

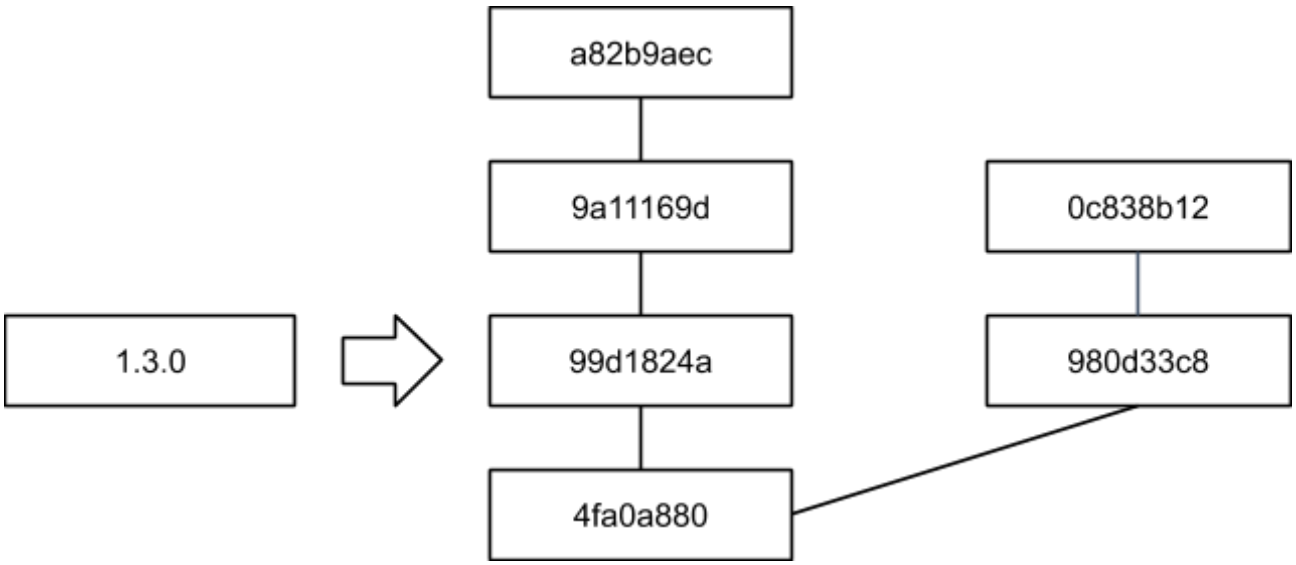
В рамках патч-версии изменений в возможностях программного обеспечения не производится. Как правило, в рамках такой версии закрываются критические уязвимости.

Это не единственно возможный способ версионирования, например, можно ориентироваться на даты выхода продукта. Так поступают во многих дистрибутивах Linux, например Ubuntu, и в линейке интеграционных сред JetBrains.

Какая бы модель версионирования не была выбрана, в системе контроля версий придется отметить коммит, который отвечает за версию, чтобы в любой момент времени иметь возможность собрать нужную версию продукта.

Тэги

Для специальной пометки коммитов git предоставляет механизм тэгов или, как еще говорят, меток.



Как мы с вами выяснили в предыдущих роликах, ветки состоят из снимков проекта — коммитов. Каждый из коммитов помечается хэшем, которые представляют собой уникальный набор букв и цифр.

История изменений может включать тысячи коммитов, осуществлять поиск и навигацию по которым может быть затруднительно.

Тем не менее, нам может потребоваться быстро найти коммит, который соответствует стабильной версии приложения. Эту проблему решают метки, т. е., короткое запоминающееся название, по которому коммит легко найти среди других.

Более того, метки — это тоже тэги, которые назначаются коммитам автоматически. Одному коммиту мы можем назначить любое количество тэгов.

Создание тэга

Для того, чтобы создать тэг, можно воспользоваться командой **git tag**, после которого указывается название тэга:

```
$ git tag v1.3.0
```

При попытке создания уже существующего тэга git сообщает, что такой тэг уже существует.

Тэги можно снабжать комментариями:

```
$ git tag -a v1.4.0 -m 'Новый тэг'
```

Для этого метку тэга добавляем при помощи параметра **-a** (а маленькое), а комментарий добавляем при помощи параметра **-m** (м маленькое):

```
$ git tag
v1.3.0
v1.4.0
```

Для просмотра списка доступных тэгов можно вызвать команду **git tag** без параметров. Чтобы отфильтровать вывод, можно воспользоваться поиском по шаблону:

```
$ git tag -l v1.4.*
v1.4.0
```

Символ звездочки означает любое количество других символов. Если потребуется удалить тэг, команде **git tag** следует добавить параметр **-d**:

```
$ git tag -d v1.3.0
Deleted tag 'v1.3.0' (was 089a6e8)
```

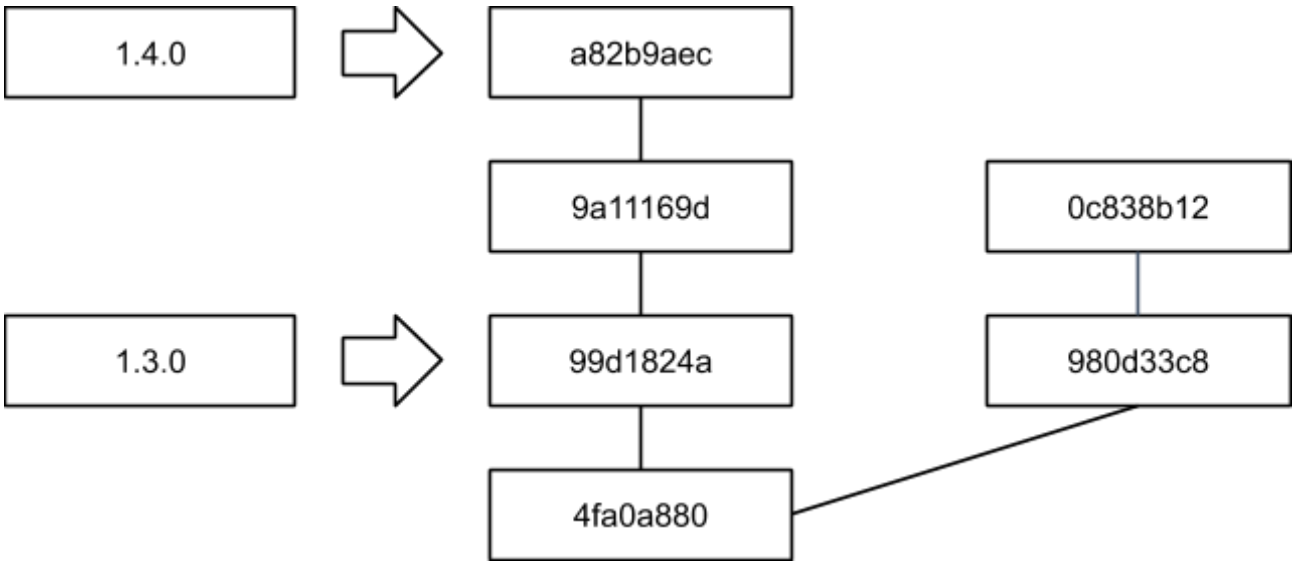
Убедимся, что тэг исчез, для этого воспользуемся командой **git tag** без параметров:

```
$ git tag
v1.4.0
```

Как видим, тэга v1.3.0 больше не существует.

Связь коммита и тэга

Тэг — это всегда ссылка на коммит. На один и тот же коммит можно навесить множество тэгов. Более того, хэш тоже можно рассматривать как тэг, просто его сложнее запомнить. Поэтому мы вводим свои собственные тэги, которые человеку запомнить гораздо проще.



Посмотрим историю коммитов при помощи команды, которую мы сконструировали из команды **git log** на одном из прошлых уроков:

```
$ git gg
* 089a6e8 - (HEAD -> master, tag: v1.4.0) New line in hello.php (Igor Simdyanov)
05.05.2019 13:15
*   316d747 - (origin/to_delete, origin/master, origin/HEAD) Merge branch 'master' of
github.com:igorsimdyanov/hello (Igor Simdyanov) 05.05.2019 12:00
...
```

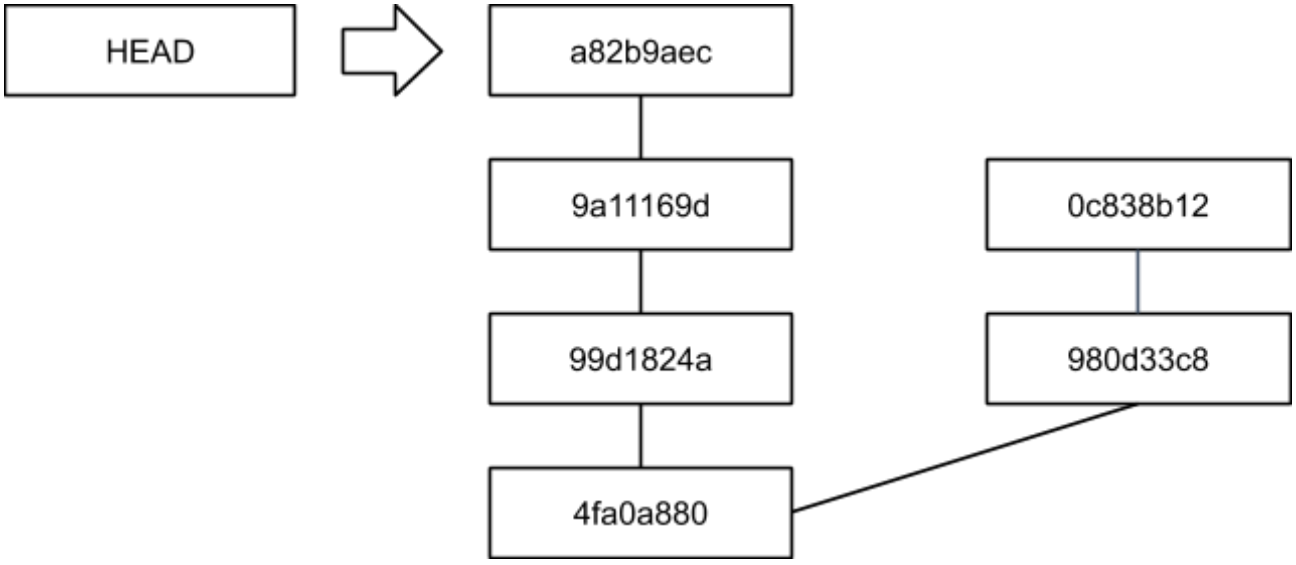
Посмотреть описание текущего коммита можно при помощи команды **git show**:

```
$ git show v1.4.0
tag v1.4.0
Tagger: Igor Simdyanov <igorsimdyanov@gmail.com>
Date:   Sun Sep 8 22:53:14 2019 +0300

Новый тэг

commit 089a6e840cdf0f00fdd4f3b385ee4f80f5329583 (HEAD -> master, tag: v1.4.0)
Author: Igor Simdyanov <igorsimdyanov@gmail.com>
Date:   Sun May 5 13:15:45 2019 +0300

      New line in hello.php

diff --git a/hello.php b/hello.php
index acfe84a..258a3a5 100644
--- a/hello.php
+++ b/hello.php
@@ -1,3 +1,4 @@
 <?php
      session_start();
      echo "Hello, world!";
```

```
+      echo "Hello, git!";
```

Вместо тэга v1.4.0 можно использовать хэш:

```
$ git show 089a6e
```



Тэг HEAD

Помимо наших собственных тэгов и хэш-кодов, git предоставляет несколько дополнительных тэгов. Один из них HEAD, указывает на последний коммит:

```
$ git gg
$ git show HEAD
```

Как видим, мы всегда можем использовать тэг HEAD для того, чтобы сослаться на последний коммит.

Переход к произвольному тэгу

По умолчанию команда git tag устанавливает метку на текущий коммит. Однако есть возможность установить тэг на совершенно произвольный коммит. Для этого в конце команды указывается хэш коммита:

```
$ git tag -a v1.2.0 -m 'Old version' beefeb1
```

Если мы теперь посмотрим историю, можно обнаружить, что тэг v1.2.0 установлен на более ранний коммит:

```
$ git gg
* 089a6e8 - (HEAD -> master, tag: v1.4.0) New line in hello.php (Igor Simdyanov)
05.05.2019 13:15
*   316d747 - (origin/to_delete, origin/master, origin/HEAD) Merge branch 'master' of
github.com:igorsimdyanov/hello (Igor Simdyanov) 05.05.2019 12:00
|\
| * 9e02299 - Индексный файл проекта (Igor Simdyanov) 05.05.2019
11:56
* | fa38fba - Добавление phpinfo-отчета (Igor Simdyanov) 05.05.2019
11:59
|/
*   de471d5 - Разрешение конфликта слияния (Igor Simdyanov)
01.04.2019 09:30
|\
| * 48e8afc - Добавляем файл hello.txt (Igor Simdyanov) 01.04.2019 09:27
* | 13a03cc - Добавление файла hello.txt (Igor Simdyanov) 01.04.2019
09:25
* |   a201de4 - Merge branch 'test' (Igor Simdyanov) 01.04.2019 09:21
|\ \
| |/
| * ad027b2 - Новый файл branch.txt (Igor Simdyanov) 01.04.2019 09:17
* | 12bec52 - Новый файл master.txt (Igor Simdyanov) 01.04.2019 09:20
|/
* beefeb1 - (tag: v1.2.0) Adding gitignore (Igor Simdyanov) 30.03.2019 12:18
* fd076e9 - Изменения в файлах (Igor Simdyanov) 24.03.2019 13:52
* c5e5079 - Добавляем файл readme.txt (Igor Simdyanov) 24.03.2019 13:50
* 08c3469 - Новый комментарий (Igor Simdyanov) 24.03.2019 12:34
* 61c926a - Еще один комментарий (Igor Simdyanov) 24.03.2019 12:26
* e6e3fd1 - Comment (Igor Simdyanov) 24.03.2019 12:24
* 09056ae - Первый коммит (Igor Simdyanov) 24.03.2019 12:21
```

Если необходимо поработать с проектом в том виде, в котором он был на момент создания тэга v1.2.0, можно отпочковать ветку от коммита, на который указывает этот тэг:

```
$ git checkout -b old_version_of_project v1.2.0
Switched to a new branch 'old_version_of_project'
```

Для этого после команды создания ветки указывается тэг. Вместо тэга можно использовать хэш коммита.

Теперь мы находимся в новой ветке, которая отпочкована от коммита, на который указывает тэг v1.2.

Отправка тэгов на удаленный репозиторий

По умолчанию команда **git push** не отправляет тэги на удаленный git-сервер. Для отправки тэгов необходимо явно указать его имя после команды **git push**:

```
$ git push origin v1.2.0
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 168 bytes | 168.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0)
To github.com:igorsimdyanov/hello.git
 * [new tag]         v1.2.0 -> v1.2.0
```

Используется та же команда, что и для отправки веток, только вместо названия ветки мы указываем название тэга.

Если перейти на GitHub на страницу релизов, можно обнаружить, что тэг v1.2.0 появился на удаленном репозитории.

Мы отправили лишь один тэг, но если их много и всех их необходимо отправить на удаленный репозиторий, можно воспользоваться командой **git push**, передав ей параметр **--tags**:

```
$ git push origin --tags
Enumerating objects: 13, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 12 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (9/9), 1.02 KiB | 1.02 MiB/s, done.
Total 9 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To github.com:igorsimdyanov/hello.git
 * [new tag]         v1.4.0 -> v1.4.0
```

Для того, чтобы запросить тэги на удаленном сервере, можно воспользоваться командой **git ls-remote**:

```
$ git ls-remote --tags origin
61be6e771aa5e1eb76e64854f4182b4a884369e4    refs/tags/v1.2.0
beefeb1c17a04efbee1a4d4a132ce767a8bde823    refs/tags/v1.2.0^{}
8eb2dd83624a912348cfdba1954b85612723cfd1    refs/tags/v1.4.0
089a6e840cdf0f00fdd4f3b385ee4f80f5329583    refs/tags/v1.4.0^{}
```

Мы получили список тэгов удаленного репозитория, не посещая GitHub. Для того, чтобы удалить тэг на удаленном сервере, можно поступить так же, как и в случае ветки.

```
$ git push origin :v1.2.0
To github.com:igorsimdyanov/hello.git
 - [deleted]         v1.2.0
```

Используем команду **git push**, после которой указываем название удаленного репозитория, в данном случае orign, и название тэга, которое предваряем двоеточием:

```
$ git ls-remote --tags origin
8eb2dd83624a912348cfdba1954b85612723cfd1    refs/tags/v1.4.0
089a6e840cdf0f00fdd4f3b385ee4f80f5329583    refs/tags/v1.4.0^{}
```

Если мы запросим список удаленных тэгов, убедимся, что тэг v1.2.0 исчез на сервере.

Используемые источники

Для подготовки методического пособия мы использовали эти ресурсы:

- Официальная документация Git: **https://git-scm.com/book/ru/v2.**