## Термины

Репозиторий - все файлы проекта, находящиеся под контролем, вместе с историей их изменения и другой служебной информацией. Специальное хранилище файлов и папок проекта, изменения в которых отслеживаются.

Индекс в Git — это специальная промежуточная область, в которой хранятся изменения файлов на пути от рабочей директории до репозитория. При выполнении коммита в него попадают только те изменения, которые были добавлены в индекс.

Один коммит — это пакет изменений, хранящий информацию с добавленными, отредактированными или удалёнными файлами кода.

Head – это указатель на текущую ветку, которая, в свою очередь, является указателем на последний коммит, сделанный в этой ветке.

В коммите содержится описание тех изменений, которые пользователь вносит в код репозитория. В Git коммит состоит из нескольких объектов. Можно считать, что коммиты это односвязный список, состоящий из объектов, в которых содержатся изменённые файл и ссылка на предыдущий коммит. Есть так же свойства даты, автора, комментарий и т.п.

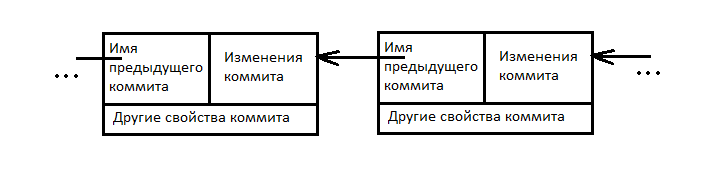


Рисунок 1 - список коммитов

Ветка – это набор коммитов, расположенных в хронологическом порядке. Ветка берёт своё начало от какого-то одного коммита. По сути ветка – это ответвление от текущей версии проекта. Все изменения, вносимые разработчиком, будут сохраняться в ветке и когда функция полностью будет реализована и проверена, можно будет слить все изменения с другой веткой.

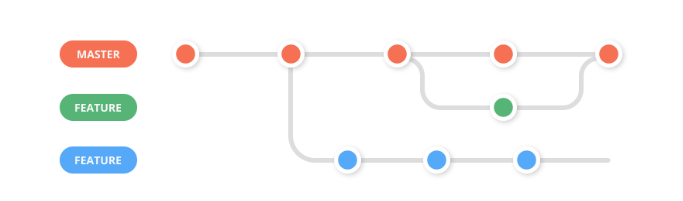


Рисунок 2 - изображение веток git

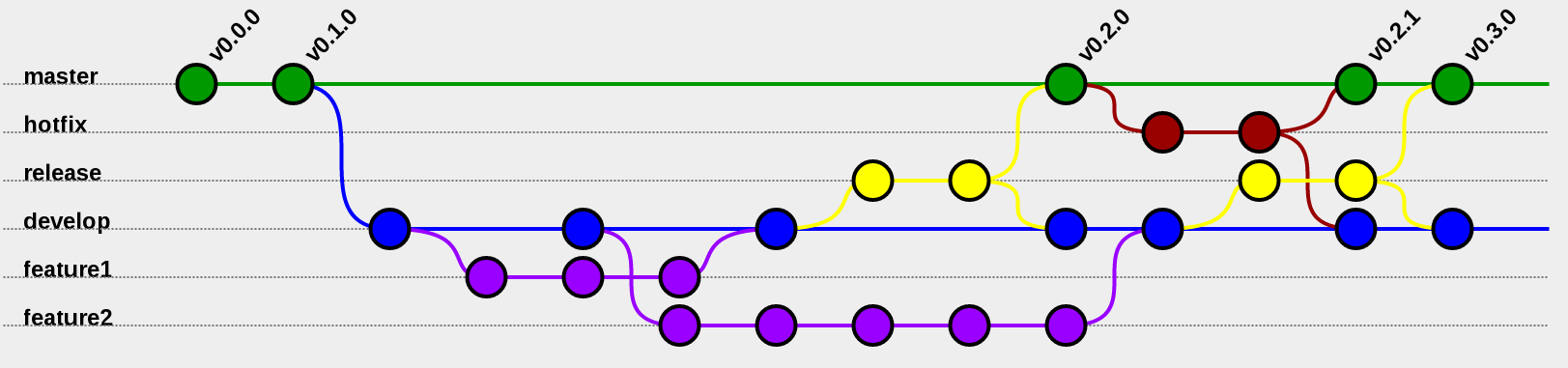


Рисунок 3 - изображение веток git

Push – это отправка данных на сервер в репозиторий.

Pull – это скачивание данных с сервера в локальный репозиторий. Pull от Clone отличается тем, что clone делаю для скачивания репозиторий с нуля. Pull для обновления данных репозитория в который уже есть локально и нужно обновить файлы.

Файл HEAD содержит название ветки или идентификатор коммита.

HEAD в git show HEAD имеет достаточно простое значение: он резолвит текущий коммит, для которого выполнен checkout! Git резолвит HEAD одним из двух способов:

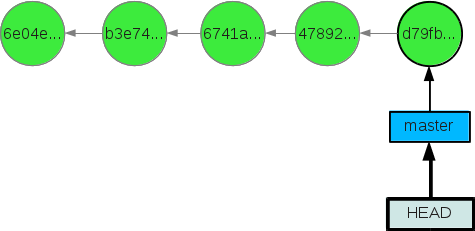
1. Если .git/HEAD содержит имя ветви, он будет последним коммитом в этой ветви (например, при чтении его из .git/refs/heads/main)
2. Если .git/HEAD содержит ID коммита, это будет ID коммита

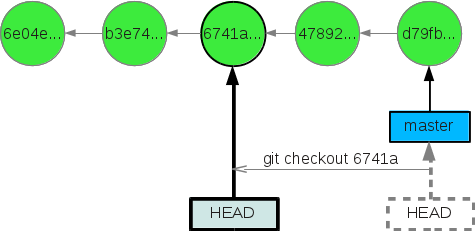
«HEAD is detached» или «detached HEAD state» означает, что отсутствует текущая ветвь.

1. commit 96fa6899ea (HEAD -> main) означает:
   1. .git/HEAD содержит ref: refs/heads/main
   2. .git/refs/heads/main содержит 96fa6899ea
2. commit 96fa6899ea (HEAD, main) означает:
   1. .git/HEAD содержит 96fa6899ea (detached HEAD)
   2. .git/refs/heads/main также содержит 96fa6899ea
3. commit 96fa6899ea (HEAD) означает:
   1. .git/HEAD содержит 96fa6899ea (detached HEAD)
   2. .git/refs/heads/main или содержит ID другого коммита, или не существует

<https://habr.com/ru/articles/800003/> - про HEAD в git

git checkout, команда которая переключает ветки, именно переносит HEAD указатель на другой коммит. Ветка – это указатель на последний коммит в выбранном разветвлении, а уже сам коммит указывает на предыдущий, тот на предыдущий и так далее. Когда я переключаю ветки я переключаю именно HEAD.





После checkout и переключения на третий коммит мы будем в состоянии файлов на два коммита назад. Если сделать git log то выдаст историю последних трёх коммитов, потому что каждый коммит содержит ссылку на предыдущий коммит и о новых коммитах он не знает.

Можно написать git log –all он напечатает всю историю коммитов.

<https://habr.com/ru/articles/157175/> - передвижение HEAD, разница с git diff

Ключ -u (полный вариант --set-upstream ) создаёт в удалённом репозитории ветку, соответствующую локальной и связывает их. Эта команда используется для отправки закоммиченных файлов в удаленный репозиторий (также известный как GitHub) в указанной ветке.

Для удаления файла из индекса и рабочей копии:

git rm –cached <название файла> удалит файл из индекса

git rm <название файла> - удалит файл из рабочей области и из индекса

Для восстановления файла в индексе и рабочей директории:

git restore –staged –worktree <имя файла> - восстановит и в индексе и в рабочей директории

git restore –source <хэш коммита> <название файла> - восстановит состояние файла на момент выбранного коммита

Переместить HEAD указатель можно следующими способами:

Указывая путь до чего бы то ни было, вы можете использовать как абсолютные указатели, например, хэш коммита или имя ветки, так и относительные. Вспомним, как пользоваться относительными указателями:

1. Знак ^ означает "предыдущий". Например: путь HEAD^ означает "предыдущий коммит перед тем, на который указывает HEAD"
2. Знак ~ позволяет вам указать число коммитов. Например, запись HEAD~7 означает "7 коммитов назад от коммита, на который указывает HEAD".

Источник упражнений по гиту и перемещению HEAD: https://smartiqa.ru/courses/git/lesson-4#theory

## Команды

`git init` - инициализация репозитория

`git add [fileName]` - делает файл готовым для коммита(добавляет файлы в индекс).

`git add .` - делает все файлы готовыми к коммиту

`git commit -m “something text”` - команда для создания коммита, после -m идёт комментарий к коммиту, его описание. Команда фиксирует все изменения в коммите.

`git log` - выводит историю коммитов. Сверху вниз, от самого нового до самого старого.

`git branch` - список веток и выбранная ветка

`git branch [new\_branch\_name]` - создание новой ветки с заданным именем

`git checkout [branch\_name]` - переключиться на ветку

`git diff` - нужна для сравнения веток или файлов

`git show` - показать какой-либо объект, ветку, файл, коммит и тд

`git add --patch` - частичное добавление файлов в индекс